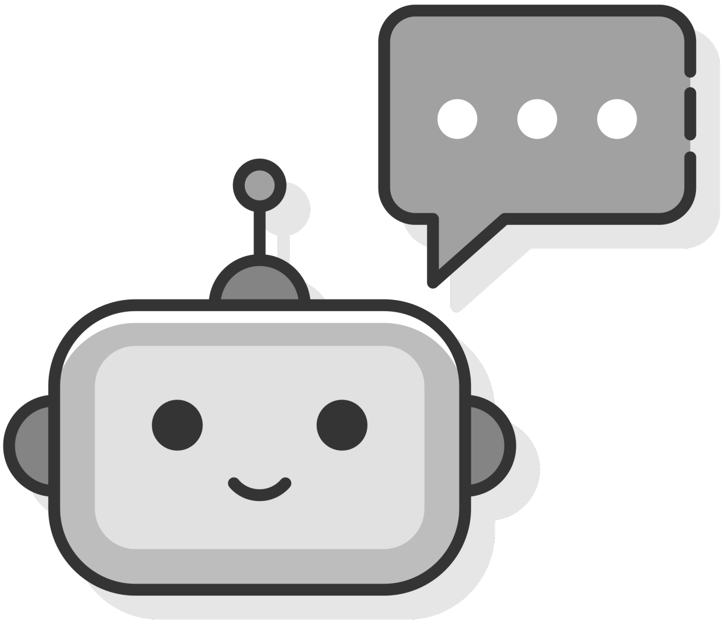
Building Interactive Text-to-SQL Systems

Version of May 10, 2022



Reinier Wessel Koops

Building Interactive Text-to-SQL Systems

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Reinier Wessel Koops
born in Hillegom, the Netherlands

Web Information Systems Research Group
Department of Software Technology
Faculty EEMCS,
Delft University of Technology
Delft, the Netherlands
www.ewi.tudelft.nl

AI for Fintech Research Lab
ING Analytics |Data, Tools &
Technology
Frankemaheerd 1
Amsterdam, the Netherlands
www.ing.nl

Cover picture: Icon made by Flat Icons from www.flaticon.com.

Building Interactive Text-to-SQL Systems

Author: Reinier Wessel Koops
Student id: 4704312
Email: R.W.Koops@student.tudelft.nl

Abstract

Natural Language Interfaces for Databases (NLIDBs) offer a way for users to reason about data. It does not require the user to know the data structure, its relations, or familiarity with a query language like SQL. It only requires the use of Natural Language. This thesis focuses on a subset of NLIDBs, namely those with ‘plain English’ sentences as input and SQL queries as output.

Study 1 recruits participants from multiple origins (i.e. academia, a crowdsourcing platform, banking industry) without selection based on their query language capabilities. Next, participants are segmented based on query language capabilities to distinguish between non-experts and experts. A common way to retrieve information from databases is by using SQL. Thus knowledge of SQL is assumed to be a proxy for participants’ skill level (i.e. SQL proficient, non-SQL proficient). We create an approach that uses an automated near semantic equivalence evaluation for user-generated queries against a predefined gold-standard SQL query and thus segment participants. We find that 70 out of 242 participants are identified as SQL proficient. To differentiate between the segmentations, we define 42 requirements often implemented for NLIDB systems, from which both segmentations pick a selection as their preferred requirements. We are unable to find statistically significant differences between the segmentations’ preferences. However, exploratory findings reveal the importance of origin, namely the banking industry, which prefers explanation over answer accuracy, different from other segmentations.

Study 2 is inspired by the exploratory findings of Study 1 and uses requirements from Study 1 to create an application that tests two conditions, one with an explanation by using color-coding (i.e. to show the relations between the natural language question asked and the models’ output columns) and another without. NLIDBs make it hard for users to verify if the answer provided by its model is correct. Therefore, Study 2 uses these two conditions above to test if color-coding improves performance for the participants. Our findings suggest that color-coding only improves performance for non-aggregate selection queries with multiple columns.

Thesis Committee:

Chair:	Prof. Dr. Ir. G. J. P. M. Houben, Faculty EEMCS, TU Delft
University supervisor:	Dr. U. K. Gadiraju, Faculty EEMCS, TU Delft
Company supervisor:	J. Brons, Chapter Lead, ING
Committee Member:	Dr. G. Lan, Faculty EEMCS, TU Delft

Preface

I want to thank my supervisor Dr. Ujwal Gadiraju, daily supervisor Sara Salimzadeh, and ING supervisor Jerry Brons for their guidance amidst the COVID pandemic. Their advice, knowledge, support, and ease of communication made our (primarily online) meetings a great way to learn and an enjoyable experience overall. It helped me stay motivated and improve the results.

Furthermore, I would like to thank the AFR lab for the weekly meetups, gatherings, and overall pleasant experiences during my thesis.

I would also like to thank my colleagues at ING, Ph.D. students from the TU Delft WIS group, and acquaintances for their insights and willingness to participate in Study 1. In particular, I would like to thank Elvan Kula, Georgios Siachamis, and Tim Draws for their advice and help.

Next, I want to thank Prof. Dr. Geert-Jan Houben and Dr. Guohao Lan, for joining the thesis committee and for their interest in my work.

Lastly, I want to thank my girlfriend, parents, and friends for their support during my thesis.

Reinier Wessel Koops
Delft, the Netherlands
May 10, 2022

Contents

List of Figures

Chapter 1

Introduction

Significant parts of structured data generated globally are stored in relational databases. Information from these relational databases is retrieved with Structured Query Language (SQL). These databases are used for many types of applications. Examples are banking, accounting, customer management software, and mobile phone apps. Retrieving such information often requires knowledge of how to query such data. The declarative approach of SQL, combined with its syntax, poses a challenge for users lacking expertise in query languages trying to retrieve information from such a relational source.

Therefore recent developments in Deep Learning (DL) for Natural Language Interfaces for Databases (NLIDBs) have increased interest in building systems that require no knowledge of the database schema structure and query language. NLIDB users only need to type their request in Natural Language ('plain English'), which is then translated to SQL. Thus, NLIDBs add an abstraction layer to make querying data easier and more accessible. It also adds the ability to reason about structured data. When we look for an answer via existing search engines, the answer is found by searching through documents of a knowledge base and returning records or excerpts detailing the most relevant parts of the records as an answer. NLIDBs map the users' natural language request into an SQL statement. Doing so can add a new layer of meaning to structured data. So instead of retrieving records, we can retrieve data from multiple records (i.e. tables), combine them and thus create new meaning by way of join operations and aggregation operations, among others. Lastly, it could provide query suggestions to the user and let the user use those suggestions as a starting point for writing SQL queries.

Due to its increased interest there are many implementations that can be found at the leaderboards of datasets like Spider (Yu et al., 2018), SParC (Yu, Zhang, Yasunaga, et al., 2019), CoSQL (Yu, Zhang, Er, et al., 2019) and WikiSQL (V. Zhong et al., 2017). These often focus on increasing the performance metric rather than on how to interact with its users. This introduces a research gap: a limited understanding of the NLIDB users. Only one instance containing a user validation study has been found (Narechania et al., 2021).

1.1 Research Questions

In this thesis, crowdsourcing is used to perform two user studies. Each study provides answers to different user-related research gaps found in the literature.

1.1.1 Study 1

There are many NLIDB implementations for the previously mentioned datasets. Often there is an overlap of requirements between implementations. Their supposed user group is often described as "[...] users who are not proficient in query languages" (Wang et al., 2020). However, this excludes users proficient in query languages as potential users, while these may also want to use NLIDBs, although supposedly for different use cases. No user study was found in the literature that evaluates either group's preferences and their supposed differences.

Thus for Study 1, we define two segments for NLIDBs, namely: **SQL proficient** and **SQL non-proficient**, where SQL acts as a proxy for query language knowledge. This segmentation of user groups also implies a difference between these user groups. So our hypothesis states that users' preferences will be different for each segmentation. From this hypothesis follows our research questions used to investigate this premise:

- **RQ1: How can different user groups (SQL proficient and non-SQL proficient) be identified, based on literature and dataset outcome?**
  The participant's user group is based on their ability to write SQL queries during the study. An open-source approach is created as described in Section 3.4.2 which evaluates these written SQL queries automatically for each participant. This approach identifies how these different user groups can be identified.
- **RQ2: What differences in preferences can be observed between the identified user groups?**
  Different requirements are identified, which are based on literature as described in Section 3.2. These are thematically grouped, where from each group, a participant selects their preferred requirement. These preferences are explored in Section 3.5. Accordingly, the hypotheses stated in the previously mentioned section are presented in Table 3.6, which displays these twenty explored hypotheses. The findings are further discussed in Section 3.6.

Contributions

Study 1 aims to address knowledge gaps in the field of NLIDBs. These can be summarized as the following contributions:

1. An overview of common requirements found in the literature regarding NLIDBs.
2. An open-sourced approach[1] of automatically evaluating the (approximate) semantical equivalence of SQL queries written by users in which previous research is combined and extended into one approach.

[1] https://doi.org/10.4121/19733029

3. A quantitative assessment between SQL proficient users and SQL non-proficient users regarding their preferred requirements.
4. A publication of all the gathered data, anonymized[2].

1.1.2 Study 2

For Study 1, we could not conclude that there is a difference between the preferred requirements of the segmented groups. However, Study 1 recruits participants from different origins (i.e. a crowdsourcing platform, banking industry, and academia). So, exploratory analysis segmenting participants by their origins revealed that the banking industry had, different from the groups from our previous and current segmentations, a preference for explainability rather than performance of the NLIDB model. This finding, together with a related study by Narechania et al., 2021, was the inspiration for Study 2. The related study was the only found user study regarding the explainability of NLIDBs.

Currently, the accuracy of NatSQL averages around 73% and drops to around 52% when the difficulty of the SQL query increases (Gan et al., 2021). This is, at the time of writing, ranked second on the Spider leaderboard, a commonly used dataset for NLIDBs (Yu et al., 2018). Thus to help an NLIDB user assess the answer provided by the model, we build a prototype that incorporates an explainability technique. This explainability technique uses color-coding and is shown to half of the participants of Study 2. The Color-coding displays the relation between the question asked to the NLIDB model, the data the NLIDB model uses, and the outcome generated by the NLIDB outcome. The hypothesis is that using Color-coding leads to increased accuracy and lower interaction time than not using it. The increased accuracy and lowered interaction time are also described as a performance. Thus our research questions state:

- **RQ3: How does Color-coding influence the performance per query?**
- **RQ4: How does the combination of Color-coding and the SQL category influence the performance per query?**
- **RQ5: How does the SQL category influence the performance per query?**

Two independent variables are identified for Study 2: condition (Baseline or Color-coding) and SQL categories (Easy, Medium, Hard, Extra-hard). These are explored individually and in a combined setup as described in Section 4.4 via six hypotheses, using a two-way ANOVA. The findings are summarized in Section 4.6 and are further discussed in Section 4.7.

Contributions

Study 2 addresses some more NLIDB knowledge gaps, which are summarized as the following contributions:

[2] https://doi.org/10.4121/19733020

1. An open-source modification of the RAT-SQL (Wang et al., 2020) model, which exposes the relations between the question and the outcome[3].
2. An overview of which cases Color-coding can be useful.
3. A publication of all the gathered data, anonymized[4].

1.2 Outline

Chapter 2 presents background knowledge and related work of NLIDBs. Next, Study 1 is shown in Chapter 3, and Study 2 is presented in Chapter 4. These two studies mainly share the same subchapters like setup, variables, statistical hypothesis testing, data preparation, results, and discussion. But these studies are different in approach. The setup contains information regarding the study procedure, participants, relevant variables, and study implementation. Data preparation is concerned with how the data is cleaned and preprocessed and which hypothesis testing is applied. Results address the findings of the study. In discussion these findings are addressed by providing context to these results. These studies are followed up by Chapter 5: Conclusion and future work, which answers the research questions with an emphasis on the contributions of this thesis and proposes future directions for research on this topic.

[3] https://github.com/ReinierKoops/rat-sql/tree/inferQuestionWithAttention

[4] https://doi.org/10.4121/19733020

Chapter 2

Background and Related Work

2.1 Natural Language Interfaces for Databases

NLIDBs can translate a user request from Natural Language into a database query language. Ideally, users do not have to know database query language or the underlying database schema to retrieve information from it. NLIDBs is a term covering a wide variety of system types as described by Affolter et al., 2019.

Older NLIDB system types did not scale well and were difficult to use. For example, having to write requests in a rigid format, make use of keywords, limited SQL syntax support, or it would only work for one specific database schema. However, four recent datasets; WikiSQL (V. Zhong et al., 2017), Spider (Yu et al., 2018), CoSQL (Yu, Zhang, Er, et al., 2019) and SparC (Yu, Zhang, Yasunaga, et al., 2019) changed this perspective by introducing relatively large datasets and deep learning techniques (Katsogiannis-Meimarakis & Koutrika, 2021).

2.1.1 Deep Learning Approaches for NLIDBs

DL networks, in combination with these four large new datasets, allow for the identification of patterns and viewing of the task at hand from a more holistic perspective (Őzcan et al., 2020). Harnessing this technological shift enables a more natural way of accessing data. These approaches, often also called NL2SQL or Text-to-SQL, focus on translating Natural Language sentences into SQL statements. There is also related research investigating NoSQL solutions like (Mondal et al., 2019), which is not the focus of the thesis.

While these DL approaches are more scalable, it still faces challenges. Natural language is ambiguous, meaning that words can have multiple meanings (lexical ambiguity (Katsogiannis-Meimarakis & Koutrika, 2021)). Also sentences can have various interpretations (syntactic ambiguity (Katsogiannis-Meimarakis & Koutrika, 2021)), linking the database schema to the right words (schema linking), which cannot always be solved by text matching because sometimes a column might have a non-sensical abbreviation. Then there is a gap between the vocabulary used by the user and on which the system is trained on. The users' vocabulary might also contain mistakes, making closing this gap even harder. Users will also expect that if the solution works a certain way on one database, it should work similarly well on the other database (database generalization) and support multiple languages. Lastly, validating the obtained answer is tricky because the user might not know SQL, the data, and the database schema. Then confirming that the answer is correct might be difficult, which is what Study 2 looks into.

Katsogiannis-Meimarakis and Koutrika, 2021 identifies three types of DL approaches, each with a different way of handling these challenges. However, they all make use of the encoder-decoder architecture (Cho et al., 2014). Sequence-to-sequence approaches attempt to transform an input query into an output query but disregard the strict grammar rules of SQL. Therefore, grammar-based approaches, like RAT-SQL (Wang et al., 2020) used for Study 2, are essentially an evolution of the sequence-to-sequence approaches because these grammar rules are now part of the transformation from input to output. Lastly, sketch-based slot-filling approaches try to simplify the generation of the SQL query to an easier predefined form, where the structure of the query is already defined. However, the variables need to be filled in. This approach is often unable to handle complex SQL queries.

2.1.2 RAT-SQL

Study 2 makes use of RAT-SQL (Wang et al., 2020), a grammar-based approach that makes use of an encoder technique called relation-aware self-attention mechanism, that encodes the database schema, performs schema linking and handles feature representation.

The technique called “Attention” was introduced as a way to improve RNNs, a sequential way of processing sequence “translation”, in our case from NL to SQL (Bahdanau et al., 2014). By adding Attention, it could better encode the context (relations) of the sequence (input) to sequence (output) translations by allowing the model to reflect better relations of each part of the input sequence with each part of the output sequence. Self-attention, which is also called intra-attention, is an extension of attention (Vaswani et al., 2017). This extension allows encoding sequences without using RNNs by introducing attention mechanisms. These mechanisms better capture how each sequence relates to the next sequence.

Schema encoding encodes all parts of the database schema, like columns and tables, in a format suitable for the model to be used during translation.

Schema-linking links words from the input to parts of the database schema. Thus, the focus is two-fold: schema relations and the input context.

Bogin et al., 2019 encoded its schema encoding using a graph neural network (GNN), where the schema encoding and linking were performed as a separate process. This is where RAT-SQL differs; it encodes these two concepts together by using relation-aware self-attention (Wang et al., 2020). This encoding uses a graph representation of the database schema, tables, columns, and the input as nodes and their relations as edges. The edges between DB schema elements are based on their natural relations, but the input uses the DB schema text matching, which we call schema linking. Accordingly, Study 2 employs RAT-SQL, using these codified relations as an explanatory mechanism. This explanation mechanism falls into the category of Local Self-Explaining, as described by Danilevsky et al., 2020, since these attention features are used to visualize how the current question relates to the database schema.

2.1.3 User Interaction Category

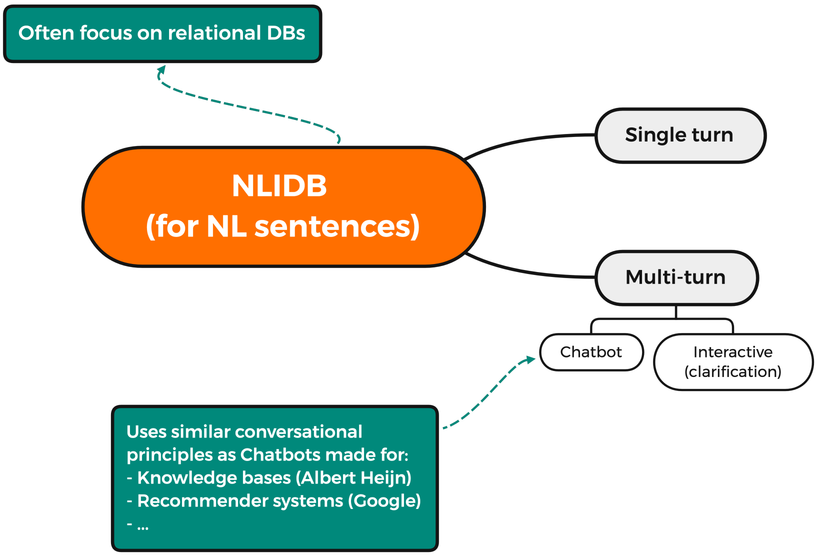

Often focus on relational DBs
NLIDB
(for NL sentences)
Single turn
Multi-turn
Chatbot
Interactive
(clarification)
Uses similar conversational
principles as Chatbots made for:
- Knowledge bases (Albert Heijn)
- Recommender systems (Google)
- ...

Figure 2.1: User interaction categorization overview of NLIDBs for Natural Language sentences.

RAT-SQL is a single turn interaction implementation as is used in Study 2 (Wang et al., 2020). This means that the interaction context with the user is only limited to the current question. As shown in Figure 2.1, there are also two other categories. A reinforcement-based implementation of MISP is an interactive multi-turn model (Yao et al., 2020). This means that the model improves based on user feedback and can thus take multiple turns to answer and correct a question. RAT-SQL and MISP are both based on the WikiSQL (V. Zhong et al., 2017) and Spider (Yu et al., 2018), which are intended only to support single turn interaction. However, some extensions support multi-turn “chatbot” (Yu, Zhang, Yasunaga, et al., 2019) (Yu, Zhang, Er, et al., 2019). This we call “chatbot” because each question takes multiple turns to complete. Therefore the model interactions should encapsulate conversational concepts like remembering previous interactions to use for current interaction as implementations like IGSQL (Cai & Wan, 2020), and R^{2}SQL (Hui et al., 2021) do.

2.2 Datasets

Both Study 1 and 2 make use of the Spider dataset (Yu et al., 2018) and its evaluation tools (R. Zhong et al., 2020). This dataset comparatively supports more SQL syntax than WikiSQL (V. Zhong et al., 2017), it has an approximate semantic evaluation tool which can be used to automatically evaluate algorithmicly generated SQL queries and has relatively more open sourced implementations than SParC (Yu, Zhang, Yasunaga, et al., 2019) and CoSQL (Yu, Zhang, Er, et al., 2019).

Approximate semantic evaluation is the only measurement currently available that approaches semantically evaluating SQL queries. Other measurements described by Kim (Kim et al., 2020) and the Spider dataset (Yu et al., 2018) like manual matching, (partial) set matching, exact string matching, parse tree matching, result matching, mathematically proving (Cosette (Chu et al., 2017)) are either limited in what they can evaluate or produce many false positives. So, this evaluation method is essential for Study 1 and 2 since this allows automated checking of SQL correctness for user-generated queries.

Even though Spider is state-of-the-art (SOTA) in the field, it has its limitations, like its focus only on Data Query Language (DQL), which is a subset of the functionality available for the SQL language (Yu et al., 2018). Also, the coverage of the DQL syntax is limited. Next, databases in practice are often extensive, but analysis on Spider revealed the median database size to be 28kb, indicating databases of limited scope and containing little data. This differs from the industry since performance optimizations can influence the kind of queries you might write. Furthermore, artificial metadata is added to the dataset containing information like full explicit column names, which in practice often is not present. In practice, you might find abbreviated column names, making schema-linking harder. Also, Spider's current implementation only supports SQLite.

2.3 User Studies

Papers about NLIDBs for sentences (e.g. SmBoP (Rubin & Berant, 2021), PICARD (Scholak et al., 2021) and RAT-SQL (Wang et al., 2020)) often focus on improving performance rather than the users of such a system. Consequently, only a few user study papers were discovered. In the paper of Yao et al., 2020, reinforcement learning is utilized using simulated user interaction. Thus no real users were used. Related to that is a dataset (Elgohary et al., 2020) and implementation (Elgohary et al., 2021) which encodes user-system interaction, where this simulated user provides feedback to the system on how to improve the NL-to-SQL-translation errors made. Next, the paper of Li et al., 2020 uses real users to improve systems' performance. Lastly, DIY by Narechania et al., 2021 performs a user evaluation study to determine if its implementation improves the comprehension of its users. It performs a limited user study using the System Usability Scale (Brooke, 1996) with a sample size of 12.

Where most of these user study papers focused on utilizing users to improve performance, DIY (Narechania et al., 2021) was the only paper found focusing on evaluating the system from a user perspective.

2.4 SQL Assessment

Most people will not be able to work with SQL. This is because "SQL is a non-trivial skill to master" (Renaud & van Biljon, 2004). As Dekeyser et al., 2007 points out that this might be due to the declarative nature of the language, which requires people to think in sets rather than steps. This imposes a hurdle on users, of which the size is dependent on their level of SQL proficiency. In literature not much has been found regarding difficulty of SQL concepts used, except the paper of Renaud and van Biljon, 2004 and Dekeyser et al., 2007. This paper by Renaud and van Biljon, 2004 states that SQL is a constructivist skill that requires users to first understand the foundations before learning the SQL concepts. These foundations mean knowledge of relational algebra, data models, set theory, and logic. Accordingly, the SQL concepts are ordered in level of supposed conceptual difficulty by Renaud and van Biljon, 2004.

Study 1 uses this order of conceptual difficulty when selecting the SQL queries and their order in the survey. These SQL queries are taken from the Spider dev dataset to be automatically evaluated, where the tools used are inspired by Kim et al., 2020. Section 3.4.2 of Study 1 elaborates on which tools are used and how they are used.

2.5 Explainable Artificial Intelligence

An NLIDB tries to find the most likely SQL prediction it can map the users' NL input into. When such a prediction is wrong, it can be helpful to provide insight to the user by explaining the process. Some implementations harness their explainability as a way for the user to provide feedback, with which the performance of the system can improve (Yao et al., 2019) (Li et al., 2020). Others offer only insight like in the paper of Narechania et al., 2021 and in Study 2.

As Došilović et al., 2018 mentions, with many of the current SOTA models, which function as a black box, there is a trade-off between predictive performance and transparency (readability). Thus leading to a lack of transparency and interpretability.

These two essential lacking parts are often required. Examples include applications in the judicial system, autonomous transport, and financial system. These sectors require algorithmic decisions to provide a solid and transparent rationale. Such a rationale acts as a proxy for trust and is thus crucial for adapting these models in these areas.

Trust requires many criteria to be met but is hard to quantify and formalize precisely as pointed out by Došilović et al., 2018. Where one such an example cares about unbiasedness (fairness), others might care more about reliability, safety, or a combination. Since it is hard to quantify trust, criteria like explainability and interpretability are often used as intermediaries to strive for.

However, as Došilović et al., 2018 points out, these criteria's definitions are dependent on how they are applied. This means it is dependent on the user's preferences, expertise, and other context-related criteria. In the context of Machine Learning, Došilović et al., 2018 regards interpretability as ”the ability to explain or to present in understandable terms to humans.

But, as the author Došilović et al., 2018 also points out, interpretability and explainability often get mixed and therefore is described that interpretability is to be model-centric: ”the mapping of abstract concepts into a domain humans can make sense of”, focusing on ”global interpretability”. Then explainability is subject-centric: ”the collection of features of the interpretable domain that have contributed to a given an example to produce a decision”, focusing on ”local interpretability”, applicable for Study 2.

Apart from the scope, it is also important to determine when such an explanation is created. Some use-cases prefer, or only allow, explanations to be given after predictions are made, while other approaches incorporate it in the prediction process (Danilevsky et al., 2020). As was stated in Section 2.1.2, the modified RAT-SQL (Wang et al., 2020) implementation of Study 2 employs attention which is an explanation technique generated locally as part of the prediction process and so is called Local Self-Explaining by Danilevsky et al., 2020.

Chapter 3

Study 1

In related work, a knowledge gap was revealed regarding the user aspect of NLIDBs, specifically NLIDBs, which translate Natural language sentences into SQL. This NLIDB subcategory seems to focus mainly on the performance of the models rather than the potential user. Only one user study was found (Narechania et al., 2021) that used the System Usability Scale (Brooke, 1996) to evaluate the user experience of their respective NLIDB with a sample size of 12.

Due to the limited user studies for NLIDB, which translate Natural Language sentences into SQL, user groups for such a system are often only assumed to be “users not experts in database querying” (Li et al., 2020), “non-technical business owners” (Őzcan et al., 2020) or a related description (Yao et al., 2019) (Baik et al., 2019) (Zeng et al., 2020) indicating a lack of query language knowledge, which excludes users that have query language knowledge.

So, given the potential of NLIDBs (e.g. using NLIDBs to auto-suggest SQL queries, its ability to reason about data) and the lack of research regarding its user groups, Study 1 was performed to investigate this research gap.

This means we want to identify users as having query language knowledge or not. Since identifying such knowledge is difficult to determine, an assumption is made. This assumption is that since datasets for this NLIDB subcategory often are relational and thus use SQL, we assume that SQL knowledge equates to query language knowledge. This means that the user groups that will be identified should be interpreted as people that know SQL or not.

A novel approach, inspired by (Kim et al., 2020), will be used to identify these user groups as described in Section 3.4.2. Users from different origins are recruited, whichever their knowledge of SQL. These user group segmentations will thus be called **SQL proficient user** and **SQL non-proficient user**. To quantify the differences between these segmentations, we identify commonly found NLIDB requirements from literature as described in Section 3.2. These requirements are thematically grouped. Participants of Study 1 are inquired about their preference regarding their preferred requirement of each grouping. This approach considers the algorithm type, communication style, and embodiment of the 7 Principles of Universal Design as described by Story et al., 1998, system modality, and supported platforms.

This chapter contains the exploratory setup, variables, statistical hypothesis testing, data preparation, results, and discussion of Study 1.

3.1 Exploratory Setup

User study participants are tasked with identifying requirements they deem most important for an NLIDB that translates English sentences into SQL. Users working with NLIDBs will have varying SQL skill levels and consequently are taken into consideration with the survey by splitting up the user group into the aforementioned **SQL proficient user** and **SQL non-proficient user**.

3.1.1 Objective

Study 1 is performed to answer our hypothesis. Namely, SQL proficient users have different Baseline preferences compared to SQL non-proficient users. First, common functional and non-functional requirements as found in literature need to be defined. This allows answering the following questions:

RQ1 How can different user groups (SQL non-proficient and SQL proficient user) be identified, based on literature and dataset outcome?

RQ2 What differences in preferences can be observed between the identified user groups?

3.1.2 Case study

Participants are tasked to identify the preferred requirements for NLIDBs. These key requirements are based on what is found in research and our survey. The questions found in the survey are (often) asked in a mutually exclusive way to identify these distinctive preferences.

Procedure

The survey has at least four parts in common for each participant. The difference between participants lies in which platform each participant joins and then is capable of performing SQL-related tasks. A visual representation of the process can be seen in Figure 3.1.

Origin. The first step for each participant is to enter via a link. This link can be distributed through six distinct ways: LinkedIn redirect, Twitter redirect, a personal invitation, TU Delft email invitation, ING email invitation, or via the Prolific recruitment platform. Participants of ING will use Microsoft Forms, while the other participants will use Qualtrics. While both platforms are commonly used, Qualtrics provides more options. Therefore the content is the same for both platforms. However, the way it is presented differs. Qualtrics allowed randomizing answers, custom CSS, and Javascript for tooltips, unlike Microsoft Forms. This was a limitation to the study since it is ING company policy to use Microsoft forms and discourage usage of other platforms. We believe the Origin factor to be a confounding variable, as described in Section 3.2.3 and Section 3.6.1. The participants were recruited based on their willingness to participate and supplemented by crowdsourced participants from the Prolific platform.

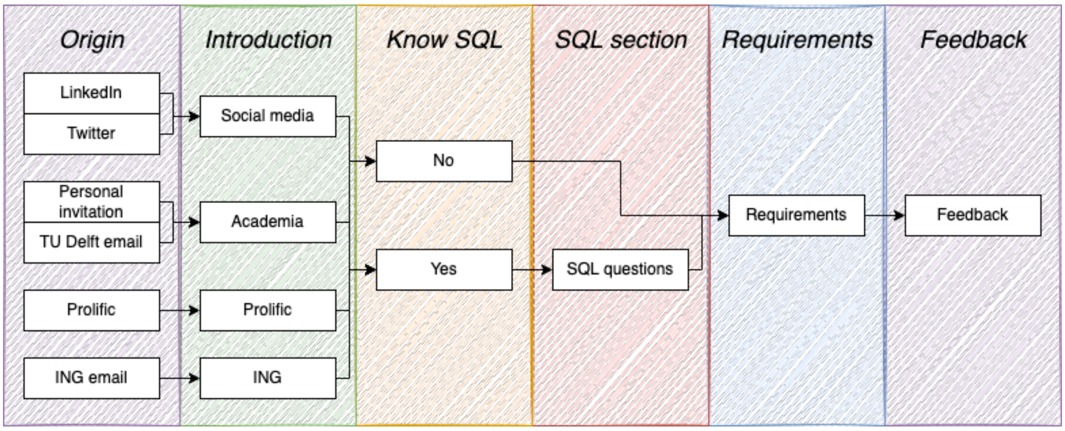


Figure 3.1: The survey flow for each participant of Study 1.

Introduction. There are four types of introductions defined for the survey of Study 1. Prolific and ING each have their separate introduction. LinkedIn and Twitter have a shared introduction, just like participants recruited via personal and TU Delft email.

Know SQL. All participants are asked if they are familiar with SQL. If they confirm to be familiar, they get objectively quantifiable SQL questions to verify this. This means requiring them to write queries related to the portrayed use case. Otherwise, they skip the SQL section.

SQL section. The group of participants that mentioned being familiar with SQL were tested via multiple SQL questions. This influences the estimated finish time from 12.22 minutes (std 9.18 minutes) to 37.09 minutes (std 47.52 minutes). These questions are arranged according to Renaud's order of conceptual SQL difficulty (Renaud & van Biljon, 2004). First, it starts with self-report questions followed by objectively quantifiable questions about set theory, SQL syntax, and writing some SQL queries.

Requirements. Now, all participants are quizzed about functional and non-functional requirements found in literature about NLIDBs.

Feedback. Lastly, all participants are provided the opportunity to leave feedback. Participants from the banking industry also get the option to leave their email so that we can contact them if we have further questions. Their feedback was qualitatively coded.

Participants Recruitment

For Study 1, participants were recruited using Twitter, LinkedIn, ING's internal mailing list, acquaintances, and Ph.D. candidates of the TU Delft Web Info Systems group. It was believed that diversifying participants' origins would increase the likelihood of finding participants identifiable as SQL proficient users. This was confirmed by the findings of the Pilot and Final study in Section 3.6. The filtering is further elaborated in Section 3.1.2. Participants of Social media and Academia are identified as the group academia, while ING is identified as banking industry.

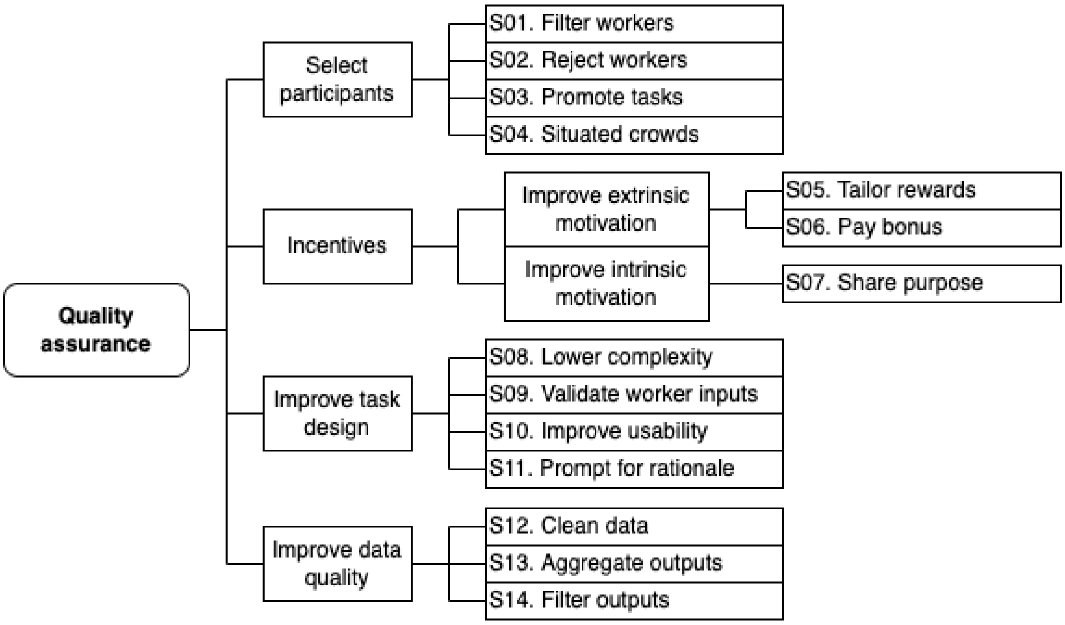


Figure 3.2: Quality assurance strategies and their corresponding measures taken for Study 1 (adapted from Daniel et al., 2018).

Quality Control To make certain that the quality of the data is high enough, 14 strategies are employed as shown in Figure 3.2. Strategies S12-S14 and S02 (Attention checks, reCAPTCHA, internal questions consistency) are described in the Section 3.4.

The participants recruited through the Prolific platform had stricter constraints (S01) for joining the survey. Their motivations for participation were believed to be mainly through extrinsic means rather than intrinsic since they are compensated financially. The rewards are tailored to the time spent and their performance (S05) and awarded bonuses accordingly (S06). This means that some participants received rewards up to £4.50 such that their hourly rate ended up to approximately £7.50 per hour. However, participants from other platforms do not get paid and therefore are intrinsically motivated to complete it. For both of these groups, the purpose of the survey is motivated (S07) in the recruitment message and introduction of the survey. This should incite a feeling of usefulness and meaning to the task. For example, in the survey, it is mentioned that it will be used for research. Another example is that participants recruited via the banking industry are specifically informed how this survey is part of the data-driven ambition of the company.

The filter on workers from Prolific (S04) is achieved by applying the following measures:

(1) **Minimum age of 18, maximum age of 50.** Prolific sets the minimum age. However, the maximum age was set to 50 since we believe participants younger are more likely to be familiar with computer software.
(2) **Fluent in English.** The survey is English.

(3) **First Language is English.** The topic of Study 1 is technical. We believe it is better to avoid non-native speakers because of potential translation issues.
(4) **Use desktop.** NLIDB are often made for use on a Desktop.
(5) **Approval rating of 95 - 100.** Higher approval rating might indicate higher quality participants.
(6) **minimum previous submissions of 50.** This ensures more advanced Prolific participants are recruited, which we believe could translate to higher quality responses.

Through message apps, emails, and social media, workers from other platforms are contacted to promote the survey (S03). Some participants were recruited physically for the pilot study to gain more direct feedback (S04).

The survey is exploratory, whereby the preferences of participants are measured. The measured preferences are most often on a conceptual level, making it hard for the participant to provide an opinion on the matter. Therefore to lower the complexity (S08) of the tasks, questions contain examples that exemplify the concept. An example is a question regarding discovery or serendipity: *"My preference when interacting with a Data Retrieval Assistant is that it focuses on finding new useful information (serendipity), finding the right information (discovery)."* Also, we replace jargon with more general words and concepts as much as possible, like in the previous example. There we describe an NLIDB as a Data Retrieval Assistant.

Validation on almost all participants' input is enforced (S09) programmatically. This means that most questions require an answer. Placeholder text and tooltips help the participant provide input that meets quality criteria.

Next, after a Pilot study, usability was improved by ensuring to abide by recommended guidelines of Qualtrics for supporting Mobile platforms (S10). Participants were provided with examples of how the answer could be answered (S11). Prompting participants for their answer rationale ensures higher data quality.

Further along, other measures were taken that do not fit the original Quality assurance strategies mentioned by Daniel (Daniel et al., 2018):

- Allow participants to provide optional additional feedback,
- Randomize question-answer order,
- Combine self-reported SQL skill level with objective knowledge measurements,
- Enforce time constraints, such that participants who perform the survey too quickly are not accepted.

We ran a pilot study to get an impression of how participants perceived and went through the survey.

Pilot and Final Study Out of the sample size of 24 valid participants, six were identified as SQL proficient users for the Pilot study as shown in Table 3.1. This limitation was because the group of expected SQL proficient users we could contact directly (academia, banking industry) was ratio-wise high but per sample low. This way (some of) the group of potential SQL proficient users we could recruit for the Final study would be bigger. For the final study, 70 were identified as SQL proficient users, a slightly higher percentage.

This pilot study helped us identify that the order of some questions should be changed and know beforehand what to expect regarding the supposed group imbalance of our sample. It also allowed us to make better time estimations for how long the study takes a participant to complete. The time estimate for SQL non-proficient users was expected to be 15 minutes and was confirmed to be true. However, this changed from 30 minutes to around 45 minutes for SQL proficient users. Also, some questions were rephrased, additional attention checks were added, as well as questions to test the internal consistency of answers, adoption of mobile-friendly styling, the addition of placeholder text and tooltips, and removal of Dutch language for the survey. Also, the pilot study helped identify that the automated SQL evaluation approach worked. Another finding was that the results indicated a linear trend regarding time completion versus the number of correctly answered SQL queries. However, this was not observed for the Final study.

	Pilot study	**Final study**	**Total**
Banking industry	3	12	15
Academia	12	21	33
Prolific	9	209	218
Total	24	242	266

Table 3.1: Participants recruited for Pilot and Final of Study 1.

Table 3.2 shows that higher scoring participants are less likely to be recruited. A higher score requires more expertise, which is less likely to be found on the Prolific platform, from which most participants were recruited. To prevent group imbalance, only two groups were considered: 0 score (SQL non-proficient user) and 1+ score (SQL proficient user).

	Correctly answered						
	0	**1**	**2**	**3**	**4**	**5**	Total
Academia	5	7	3	5	1	0	21
Prolific	161	30	15	2	1	0	209
Banking industry	6	2	1	1	2	0	12
Total	172	39	19	8	4	0	242

Table 3.2: Correctly answered per origin for Final study participants of Study 1.

3.2 Variables

The survey has multiple sections; the consent, the introductory questions, conditional SQL part, non-functional requirements part, functional requirements part, and closing questions. The features retrieved and generated from each of these parts allow us to define multiple types of variables for Study 1.

3.2.1 Independent Variables

The setup of the survey allows to segment participants in either group **SQL proficient user** or **SQL non-proficient user**. This evaluation process is elaborated in Section 3.4.2, where the answer to research question **RQ01** is explained. This segmentation of user groups acts as an independent variable and is used to evaluate research question **RQ02**. Participants that are SQL proficient have supposed different preferences than SQL non-proficient participants.

3.2.2 Dependent Variables

There are four kinds of dependent variable categories identified. These categories are Non-functional requirements relating to the algorithm of an NLIDB, Non-functional requirements relating to the conversational style of the NLIDB, Non-functional values and their importance for an NLIDB, and Functional requirements for an NLIDB. These identified requirements answer **RQ02**. We will state these as follows:

1. Non-functional requirements relating to the algorithm of an NLIDB:

    - **Self-improving or Fixed.** This trade-off is based on implementations like PICARD (Scholak et al., 2021) and MISP (Yao et al., 2019). Where PICARD is static and does not improve after being trained once, MISP is continuously improving based on user input.
    - **One shot or iterative.** These requirements were based on implementations of the Spider (Yu et al., 2018), CoSQL (Yu, Zhang, Er, et al., 2019) and SParC (Yu, Zhang, Yasunaga, et al., 2019) dataset. Where Spider dataset implementations often only answer the current question, CoSQL and SParC handle a chain of derivative questions.
    - **Serendipity or Navigation focus.** This relates to how users seek information (Foster & Ford, 2003), in this case using an information retrieving system like an NLIDB. Searching for information is often seen as discovering something new and finding the right information.
    - **Bot type**: Four bot types are identified: QA Agent, Decision Support, Task Support, Social/Chatbot. These types are described by Gao (Gao et al., 2019). The author mentions that these bot types increasingly become more intertwined and try to incorporate conversational aspects. This can also be observed by datasets like CoSQL (Yu, Zhang, Er, et al., 2019) compared to its predecessor (Yu et al., 2018).

- **Accuracy vs. Explainability.** This trade-off is based on the types of algorithms that are commonly used in the field and thus also by NLIDBs. Do you want to use Deep learning and therefore introduce a black-box algorithm that reduces explainability but increases performance, or use more traditional methods while losing performance but gaining explainability (Loyola-González, 2019)? Therefore hybrid models are an active field of research as mentioned by Őzcan et al., 2020, such that the supposed perfect balance between explainability and performance is stricken.

2. Non-functional requirements relating to the conversational style of the NLIDB:

   - **Factual/Direct or Social/Chatty.** While some users might prefer to interact with an agent that gives concise answers, others might rather have a more personable approach, akin to a ”personality”. This is based on the paper of Grudin and Jacques, 2019 which describes it can increase engagement but raise expectations significantly. When it fails, it can annoy users.
   - **Narrow, deep focus or Broad, shallow focus.** Based on the paper of Grudin and Jacques, 2019 which describes three types of chatbots, all with a different Focus. The three types are changed into two mutually exclusive types, such that making a distinction between preferences is more clear.
   - **User, Agent or mixed-initiative.** This goes into how users expect to interact with such an agent. The question is based on one of the RRIMS properties by Radlinski and Craswell, 2017, which describes the properties required for a conversational search system.

3. Non-functional values and their importance for an NLIDB rated on a five-point Likert scale (not important, slightly important, important, very important, most important). These are the 7 Principles of Universal Design (Story et al., 1998).

   - **“Equitable Use”.**
   - **“Flexible in Use”.**
   - **“Simple and Intuitive Use”.**
   - **“Perceptible Information”.**
   - **“Tolerance for Error”.**
   - **“Low Physical Effort”.**
   - **“Size and Space for Approach and Use”.**

4. Functional requirements for an NLIDB. Some of these differences are found amongst the implementations:

   - **Typing, Speech or Graphical interface modality.** Users might want to interact with the NLIDB using typing (Elgohary et al., 2021), recording their speech, or using an extensive clickable (Graphical) interface (Narechania et al., 2021).
   - **Answer, Explanation or both presented as an outcome.** Models can be used to provide only answers (Scholak et al., 2021), explain their reasoning purely, or give both (Narechania et al., 2021).

- **Mobile, Tablet or Desktop platform.** Common platforms used today for NLIDB systems are often aimed for Desktop(Zeng et al., 2020). This can be due to the limitations imposed by the Mobile and the Tablet system. These limitations can be attributed to the smaller display making it harder to display data stored in matrix-like structures. There might be more options for that in the future given the existence of datasets like ToTTo (Parikh et al., 2020), which is aimed at trying to convert matrix-like structures to text.
- **Multiple choice, Typing, Speech, Graphical or Binary choice answer types.** Currently, implementations make a distinction between asking an NL question that is translated to SQL and correcting the model if the answer (SQL) generated is wrong. PIIA corrects the model via Multiple choice (MC) (Li et al., 2020), while MISP (Yao et al., 2019) uses binary choice. NL-EDIT tries to do that through natural language (Elgohary et al., 2021), which arguably might be harder to implement. Other options are DIY, which provides a graphical interface to adapt anything (Narechania et al., 2021).

3.2.3 Potential Confounding Variables

During the survey, multiple variables were gathered. We limited the number of variables considered in our dataset regarding privacy and consistency. For every participant, we gathered the following variables:

- **Platform of origin.** Options: Banking industry, Prolific, Academia.
- **Experience with SQL.** Options: Yes, No. This is a self-reported question. If participants self-report to know SQL, they are presented with more SQL questions. Otherwise, they skip these questions.

Some of the participants are familiar with SQL, so they are represented with some more SQL questions. Thus for this group, there are a few more variables available. These variables can also be divided into a self-report and an objective assessment. The self-report contains information regarding:

- **Conceptual knowledge of SQL rules, syntax, and concepts.** Options: few SQL, most SQL, advanced SQL.
- **Years of experience using SQL.** Options: $<$1, $\approx$1, 2 - 3, 3 - 5, 5+.
- **Context of using SQL.** Options: current job, previous job, previous hobby, current hobby.
- **Self-graded SQL skill level.** Options: Poor, Fair, Good, Great, Excellent.

The rest of these variables are used to calculate or are calculated to assess the participants' skill from an objective point of view:

- **Familiarity with common SQL syntax.** Options: open answer. Expected: SELECT, FROM, and related SQL syntax.
- **Familiarity with set theory.** Options: Power set, Intersection, Union, Difference. Expected: Power set.

- **Explaining SQL outcome in NL.** Options: open answer. Expected: United Airlines.

For each of the five queries four metrics are calculated:

- **Executability of SQL query.** Options: 1.00 or 0.00. Expected: A working SQL query.
- **(Approximate) semantic equivalence.** Options: 1.00 or 0.00. Expected: Participant query matches expected query semantically.
- **Syntactic equivalence.** Options: range between 0.00 to 1.00. Expected: Participant query syntactically matches expected query.
- **Performance Score.** Options: range between 0.00 to 1.00. Expected: Participant query matches SQL query difficulty and the exact number of nested queries from the expected query.

The difficulty per SQL query is determined by order of difficulty (Renaud & van Biljon, 2004), and the type of instruction given. Each question consecutively goes from instruction-like towards a more conceptual way of asking the required information. These queries include an image detailing all the needed information from a database perspective.

The concepts used for these SQL queries are limited by what is found in the Spider dev dataset, the concepts described by Renaud and van Biljon, 2004, and the fact that the applications of the Spider dataset are focused on DQL.

3.3 Statistical Hypothesis Testing

For this study, we explored twenty hypotheses between the independent variable of SQL skill level user groups (SQL proficient user vs. SQL non-proficient user) and dependent preference variables. The survey mainly involves testing preferences, which creates non-parametric data. Therefore Chi-square tests of independence are used. Such a test can deal with categorical data containing two or more categories. When using G*Power (Faul et al., 2007) Apriori analysis, we define Effect size (w) of 0.3 (medium according to Cohen, 1988), α of 0.05 correcting for Type-I error inflation, Power ($1 - \beta$) of 0.95 correcting for Type-II errors and the maximum degrees of freedom to 4. This reveals a required sample size of 207. A sensitivity analysis revealed an effect size of 0.277 for the sample size of 242 achieved for our Final study. The family-wise error rate is corrected via the Holm-Bonferroni method (Abdi, 2010).

	H01 : H09, H17 : H20	**H10 : H16**
Statistical test	Chi square test of independence, with varying degrees of freedom	Mann-Whitney U test
Independent variable(s)	SQL skill level user group	
Dependent variable(s)	Mutually exclusive (non-)functional categories	5-point Likert scale of Importance

Table 3.3: Types of statistical tests for Study 1.

3.4 Data Preparation

3.4.1 Data Cleaning

As mentioned in the exploratory setup, Quality control measures were taken to filter out submissions not meeting the criteria. These criteria were determined before performing the survey (Figure 3.2 S12 - S14 and S02). Participants who joined the study voluntarily (banking industry, academia) were excluded from attention checks. We link the excluded submission categories to malicious behavior (Gadiraju et al., 2015) often seen when performing crowdsourcing. Gadiraju identifies five malicious types: Ineligible Workers (IE, workers who do not qualify), Fast Deceivers (FD, tried quick exploitation), Rule Breakers (RB, do not abide by the instructions), Smart Deceivers (SD, abide by the instructions but not the intention of the question) and Gold Standard Preys (GSP, fail gold standard test questions like attention checks). The criteria were as follows:

- **Failed attention check submissions.** 22 Submissions from Prolific were excluded. Prolific does not allow the exclusion of participants based on one failed attention check, given the survey takes longer than 5 minutes to complete(Prolific, n.d.). Therefore most participants with an attention check fail were accepted on the Prolific platform but excluded from the dataset. These participants were often found to exhibit malicious intent and thus are classified as **RB** however. Also, a few of these participants could be identified as **GSP** since they were not found to inhibit malicious intent for any question except for providing the wrong answers to easy attention checks.
- **Insufficient submissions.** The submissions in the following categories were excluded on the Prolific platform:
  - **Rejected.** 41 Submissions were rejected on the Prolific platform based on criteria like failing an attention check (**RB, GSP**) or being a low effort response (**FD**).
  - **Timed-out.** 6 Submissions timed out. This meant the participant either stopped the survey before completing it, experienced technical difficulties, took too long to finish, or another such unspecified reason. The study completion time was set to 14 minutes. This was based on the pilot study. Therefore the maximum time allowed by the Prolific platform for a participant is 54 minutes for completion. For these submissions, no malicious intent could be observed.
- **Low effort submissions.** The submissions pertaining to the following categories were found:
  - **Flatlining on 5-point Likert scale importance values.** 6 Submissions from all participants were excluded due to answering the same response for seven related questions, e.g., by rating all values equally important. This behavior could be classified as **SD**.
  - **Internal inconsistency on questions relating to 5-point Likert scale importance values.** 62 Submissions were excluded based on being internally inconsistent. The 5-point Likert scale importance value question asks the participant to rate multiple values according to their importance. The follow-up question

asks the participant to provide the three most important values, whereas the next question for the participant is to provide the three least important values. Whenever these questions' three most important and least important values do not coincide, a minus point is added to the entry for each wrong value. These questions take up a lot of screen space, making it likely to make a mistake, so submissions with only one value error are admissable. This means only values with zero or one value errors are accepted. These excluded submissions were identified as **RB** or **SD**.

  - **Speeding.** 5 Submissions were found to be related to speeding. These were all also identified to contain quality concerns related to attention checks and internal inconsistency. Speeding is determined according to the criteria of reading speed. Brysbaert (Brysbaert, 2019) claims an average reading speed for English non-fiction of 238 words per minute (wpm) with a standard deviation of 51.2wpm. This equates to the range of 84wpm to 392wpm. The words of the questionnaire are counted, but the time it takes to provide an answer for each question is not taken into consideration because this is context-dependent. No response time measurements were available for each question separately. Also, participants who claim to be familiar with SQL have more questions to answer. Therefore, a distinction between an **SQL proficient user** and a **SQL non-proficient user** is made to calculate the minimum time allowed for a participant. A **SQL proficient user** has to read 2202 words divided by 392wpm. Which translates to roughly 337 seconds. A **SQL non-proficient user** has to read 1610 words. This equates to approximately 246 seconds. These five submissions are therefore identified as **FD** since these submissions often were found to provide answers which were internally inconsistent and too fast to have read all the text.
  - **Incomplete.** 2 Submissions were excluded since these were found to be missing values that were required to complete the survey. The number of incomplete submissions was higher. However, the specific statistics were automatically deleted by the Qualtrics platform over time. It is estimated that there were around 80 incomplete submissions.
  - **Fraud.** 0 Submissions were found to be fraudulent according to security measures of Qualtrics (Qualtrics, n.d.). This was a bot detection mechanism using Recaptcha scores. All submissions had scores higher than 0.5, which according to Qualtrics, indicates it is likely that the participant is human.

- **Technical issue submissions.** 11 Submissions from the banking industry were removed (out of a total of 32 banking industry submissions). This was due to Microsoft Forms not programmatically enforcing participants to select exactly three options for questions 30 and 31. Even though these questions stated the required selection criteria explicitly. These submissions are therefore identified as **RB**.

In total, 144 submissions were excluded from the dataset. This total excludes the timed-out, some incomplete and speeding submissions since these were either not included in the dataset or removed via other quality criteria.

All the submissions were manually verified. This leaves the total number of valid submissions to be 243. In the process, no submissions have been manipulated.

The questionnaire leaves room for participants to provide feedback through open answer questions. Invalid and irrelevant answers were removed. The first question is related to what a Data Retrieval Assistant should be able to do. Seven answers were deemed irrelevant, like *"Retrieve information"*.

The next question was about the most important requirements for a Data Retrieval Assistant. Only one answer was removed, which was a dot. The last open question was what the participant thought was not mentioned by the questionnaire but should be considered. Seventy-seven answers were removed since these were answers stating *"I do not know"* as answers.

3.4.2 Data Preprocessing

The next step is to merge the datasets since we use two datasets from Microsoft Forms and Qualtrics, both saving their data differently, whereafter we clean the combined dataset. After that, we extract meaning from the data and add those new fields to the dataset. Each row is one unique participant response. Lastly, we perform the **S12-S14** quality assurance measures.

Merging datasets

- **Origin metric.** Each submission can be qualified into three categories pertaining to its origin. These origins are banking industry (ING), Prolific, or academia.
- **Time spent.** Every platform calculates time differently. However, every submission's total time is calculated in seconds for the merged dataset.
- **Platform specific metadata.** Removal of columns like start date, end date, recorded date, participant language, and related data was removed from the merged dataset since the metadata would potentially only have been useful if two of the same platforms were used to question participants. Also, the leftover column names from both datasets were renamed uniformly.
- **Privacy.** Response id, Prolific id, personal email, and other personal information shared for the open answer questions were all removed from the merged dataset since this information is private information that the study promised to all participants not to disclose.
- **Banking industry data.** Mentions of the word ING have been removed from the dataset and replaced by "Company".
- **Value encoding.** The values found in the columns were adapted to uniform values, be it binary, label, or one-hot encoding.

Generate SQL features

The dataset now contains queries that are yet to be evaluated. We create a new approach to assess all SQL queries created by the participants automatically. The approach is novel but related to the approach of Kim et al., 2020. Whereas that paper makes use of closed-source tools, we use open-source tools.

We have access to what we expect the query to be and the user-submitted output in this approach. Only submissions of participants who claimed to be familiar with SQL, Question 4, are considered to be evaluated by this approach.

We make use of the dev part of the Spider dataset (Yu et al., 2018), from which we select six queries. These six queries are chosen based on the syntax used. In the paper of Renaud and van Biljon, 2004, SQL concepts are presented in increasing order of complexity. SQL concepts not found in the Spider dataset are excluded. Hence these six queries contain two to three of these concepts and are ordered in increasing order of difficulty.

So, these six queries of increasing difficulty, taken from the dev dataset of the Spider dataset, are evaluated using a modified version of the Spider dataset evaluation tool (R. Zhong et al., 2020).

Figure 3.3 shows the workflow employed for evaluating SQL queries submitted by the questionnaire participants. This is described in detail:

- First, all the found SQL queries that are submitted by the participants are formatted with python-sqlparse[1] and sql-metadata[2]. Identified SQL keywords are capitalized; other words are made lowercase. Between some symbols, spaces are added to improve readability or are removed entirely. Overall this improves the automated approach. However, it also introduces false negatives regarding some case-sensitive SQL dialects. Given that this approach uses SQLite, we do not expect this to have an impact on our evaluation. But it might wrongfully impact queries intended to be evaluated for different SQL dialects. No responses were found in our dataset with this problem.
- Then, the entries are checked to see if it starts with the word ”SELECT” and if it is a ”non-malicious” SQL query. A keyword-based approach and naive SQL injection detection algorithm has been created to filter out invalid statements. This is to prevent modifications to the current database. This approach leads to a few false negatives since not all entries start with SELECT but are actually valid. Placeholder text before text entry was added to prevent such cases. No potentially malicious code was found after using this filter step when tested. In the dataset, no responses were found to contain malicious queries.
- Next, valid entries are evaluated by using something we like to call *“approximate semantic accuracy”*. For this, we make use of an adapted version of the aforementioned tool (R. Zhong et al., 2020). This tool allows us to evaluate each user-submitted query with the expected query on multiple distinct instances of the same database. Imagine it is found to produce the same output for each database with the same database schema but different data. In that case, we evaluate the query to be semantically equivalent with a Semantic score of 1.00. This also means a Correctness score of 1.00. Otherwise, the Semantic score is 0.00, and we go to the next step.

[1] https://https://github.com/andialbrecht/sqlparse

[2] https://https://https://github.com/macbre/sql-metadata

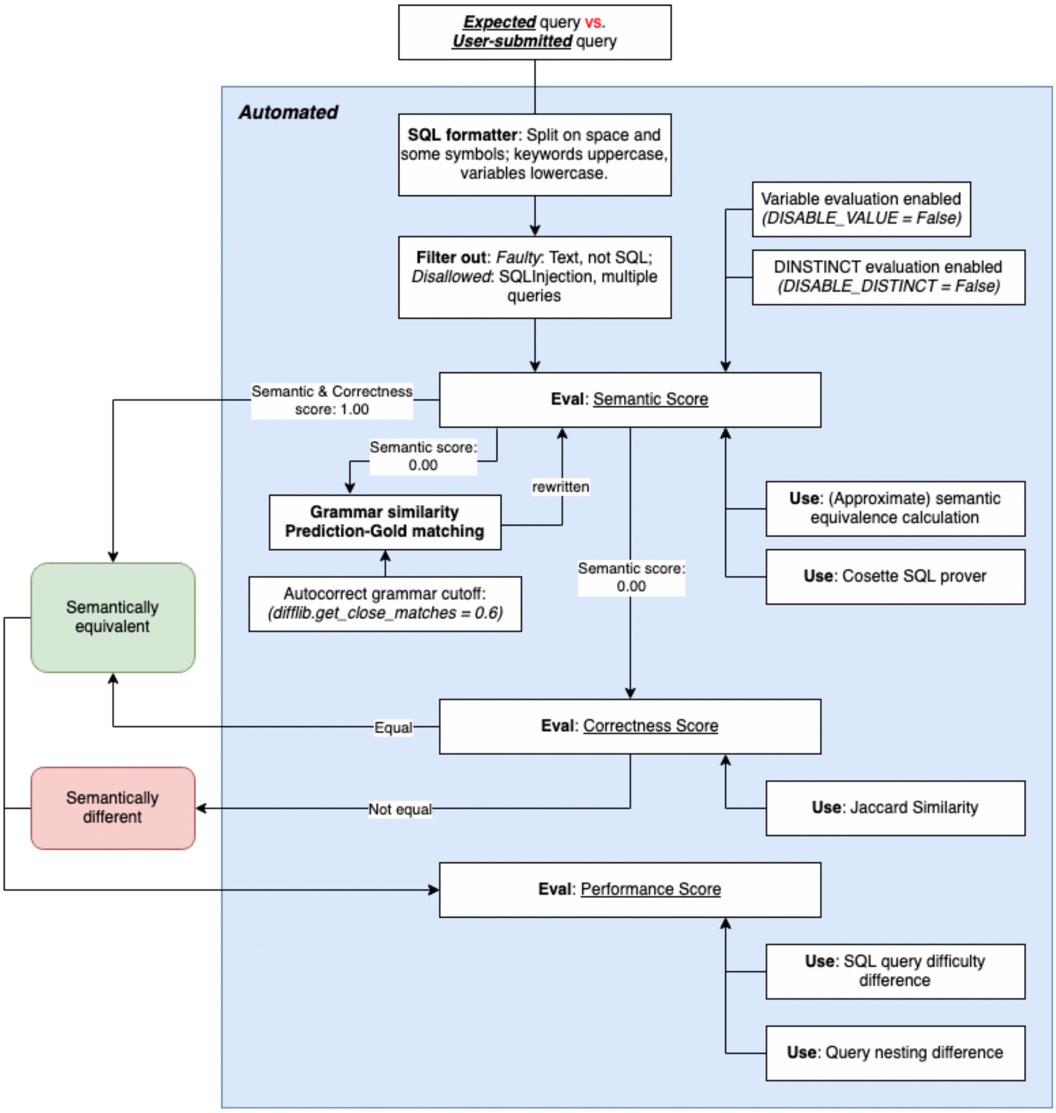

Expected query vs. User-submitted query
Automated
SQL formatter: Split on space and some symbols; keywords uppercase, variables lowercase.
Filter out: Faulty: Text, not SQL; Disallowed: SQLInjection, multiple queries
Variable evaluation enabled (DISABLE_VALUE = False)
DINSTINCT evaluation enabled (DISABLE_DISTINCT = False)
Semantic & Correctness score: 1.00
Eval: Semantic Score
Semantic score: 0.00
rewritten
Grammar similarity Prediction-Gold matching
Autocorrect grammar cutoff: (difflib.get_close_matches = 0.6)
Use: (Approximate) semantic equivalence calculation
Use: Cosette SQL prover
Semantic score: 0.00
Semantically equivalent
Equal
Eval: Correctness Score
Semantically different
Not equal
Use: Jaccard Similarity
Eval: Performance Score
Use: SQL query difficulty difference
Use: Query nesting difference

Figure 3.3: The workflow of each SQL query submitted by survey participants for Study 1.

- User-submitted queries can contain grammatical errors. To fix some of these errors, we perform a grammar similarity matching the user-submitted query with the expected query. This fixing can also introduce new mistakes when multiple words are similar. However, this only appeared to be the case for four queries. If the query changes, we perform ”approximate semantic accuracy” again. If it returns a Semantic score of 1.00, it is Semantically equivalent and has a Correctness score of 1.00. Otherwise, the Semantic score is 0.00, and we continue to the next step.
- Cosette (Chu et al., 2017) is an online tool that can prove a subset of the available SQL syntax. This means that user-submitted previously identified queries with a Correctness score of 0.00 are again tested against Cosette. Since the SQL syntax is limited,

some queries return an error because not all SQL syntax can be proven. Prematurely user-submitted queries that contain unsupported SQL syntax are excluded. This leads to only 14 queries to be run on Cosette. 5 Of these queries are found to be not equal, meaning a Semantic score of 0.00. The other nine queries cannot be evaluated using Cosette, therefor are also determined to have a Semantic score of 0.00.

- Now, we start evaluating queries on their syntax, disregarding their semantic meaning. This requires us to know all potential unique syntax present in the query. For the five employed queries, we define multiple attributes by hand based on some attribute categories defined by Kantere, 2016. These attributes are key (Ex. *"owner_id"*), select (Ex. *"winner_name"*), from (Ex. *"matches"*), all (Ex. *"winner_name", "winner_age", "matches"*), value conditions (Ex. *"1948"*) and value constraints (Ex. *"age* $<$ *30", "age* $\leq$ *29"*). Then we use Jaccard Coefficient (Niwattanakul et al., 2013) for all these attributes to calculate how close the user-submitted query is to the expected query. This is a Correctness score between a range of 0.00 and 1.00.
- Moreover, we compare all user-submitted queries that we identified as SQL queries with the expected queries for a Performance score. This score is an average of the difference found in the SQL query difficulty determined by the modified implementation of the tooling of the Spider dataset evaluation tool (R. Zhong et al., 2020) and the difference of identified subqueries (by using regex).
- Lastly, given the three scores for each user-submitted query, a participant is identified as an SQL proficient user if the user can reach a Semantic or Correctness score of 1.00 for at least one query.

This approach identified 70 out of the 242 valid participant submissions as SQL proficient users. A participant is a SQL proficient user if it has self-reported to be familiar with SQL and was able to write one SQL query out of five available correctly. The participants' answer is correct when it is (approximately) semantically or syntactically the same, having a score of 1.00.

The identification process shows a skew in the sample where only approximately 28% are identified as SQL proficient users.

3.5 Results

3.5.1 Descriptive Statistics

242 Out of 421 participants were found to be valid submissions. Out of 242, only 70 were found to be SQL proficient users. This indicates, as can be observed from Table 3.4 that the sample is unbalanced; however, given the fact that we use Chi-square tests that are insensitive to this, it proved to be no problem. There were methods of assessment in place to identify up to multiple SQL experience levels. Still, due to the difficulty of recruiting users with SQL knowledge and budget constraints, it was decided only to have two groups: SQL proficient users and SQL non-proficient users.

Platform of Origin	**SQL non-proficient user**	**SQL proficient user**	**Total**
Prolific	161	48	**209**
Academia	5	16	**21**
Banking industry	6	6	**12**
Total	**172**	**70**	**242**

Table 3.4: Identified groups combined with their platform of origin; displaying their sample sizes, Study 1.

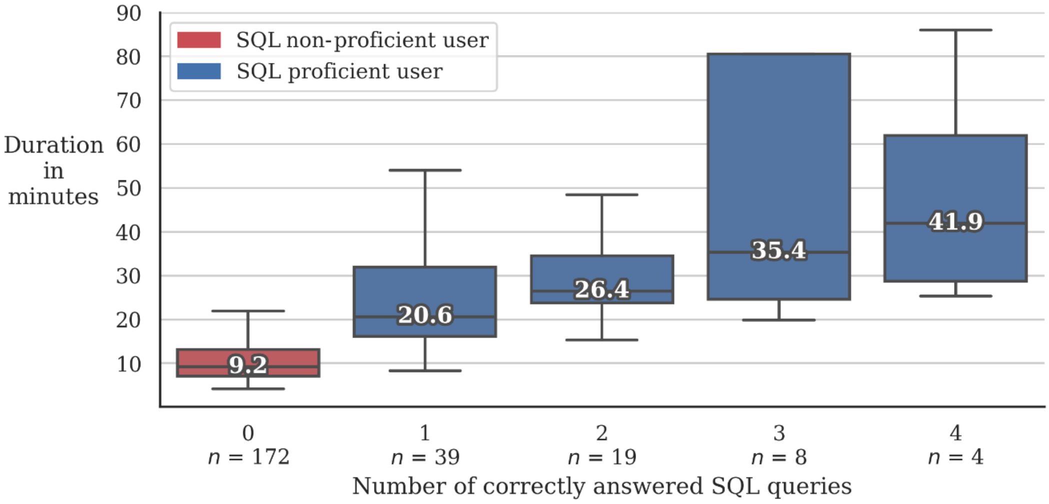


Figure 3.4: Participant time in minutes w.r.t. number of correctly answered SQL queries, Study 1.

Figure 3.4 displays the relation between SQL queries correctly answered, and the time it took to complete the survey. On average, as the score increases, so does the expected time. However, the caveat is that as the correctness score increases, the sample size decreases, which adds more uncertainty regarding the time spent, since smaller sample sizes are less reliable. Also, compared to results found during the pilot study, which showed a linear trend between time and score, the relation is much less apparent for the final study.

The average time spent on the survey was 19 minutes, 25 seconds (N = 242), with a median of 11 minutes and 48 seconds as shown in Figure 3.5. This figure also shows that if we distinguish between the user groups, there is a difference between the groups. Where SQL non-proficient users have an average time spent of 12 minutes, 13 seconds (N = 172), with a median of 9 minutes, 10 seconds, SQL proficient users have an average time spent of 37 minutes, 5 seconds (N = 70).

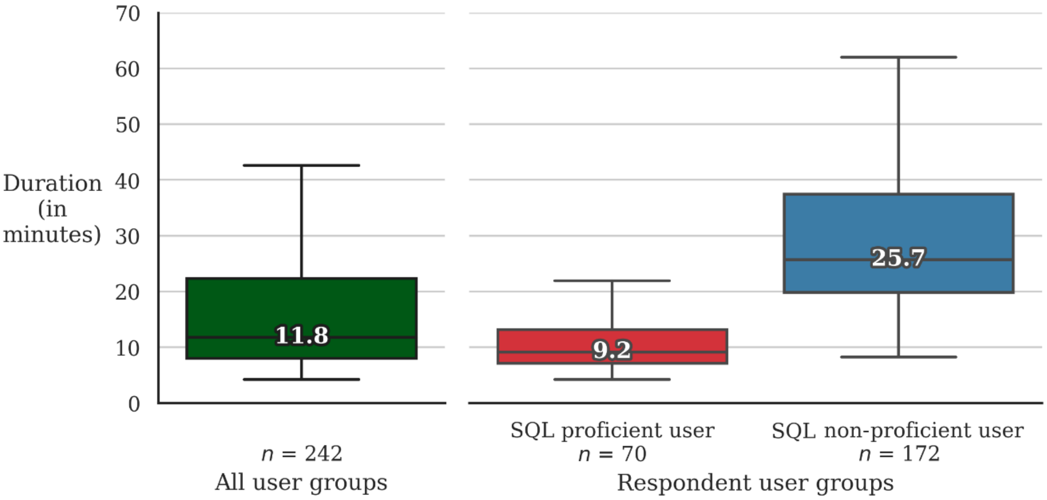


Figure 3.5: Participant time in minutes compared to user group, Study 1.

Familiarity with SQL

Users who self-reported to be familiar with SQL had their knowledge assessed via multiple questions that were based on the order of learning SQL concepts as described in the paper of Renaud and van Biljon, 2004. 157 Participants self-reported not being familiar with SQL. 85 Participants self-reported to be familiar with SQL, of which 15 participants could not be verified if they could use SQL. This leaves the total number of verified SQL proficient users to a total of the previously reported number of 70.

Out of these 15 participants, the following categories were identified:

- **13 participants**: Have knowledge of SQL, but cannot write it. Examples:
  - *"create table three youngest winner table no duplicate"*
  - *"I do not know"*
- **2 participants**: Have knowledge of SQL, but made an error. Example:
  - *"SELECT Name from singer where Birth_Year = **1944** or Birth_Year=1949"*

The participants who self-reported to be familiar with SQL were asked five self-report questions followed by eight objectively measured questions. Only five out of these eight objectively measured questions determine wheather the participant falls into the category of an SQL proficient user or not.

Self-report The self-reported SQL skill level of participants reports that SQL non-proficient users rated themselves lower (average score of 1.9, ±0.77 on a 5 point Likert scale (0: Poor, 4: Excellent)) than SQL proficient users (average score of 2.7, ±0.87 on a 5 point Likert scale (0: Poor, 4: Excellent)). Next, the same tendency was also found for the self-reported conceptual knowledge of SQL rules. Here the SQL non-proficient users reportedly had an average of 1.2 (±0.41) compared to SQL proficient users, who had 1.84 (±0.73) on a 3 point Likert scale (0: Few, 1: Most, 2: Advanced). The self-reported years of experience showed that SQL non-proficient users also had less experience (median score of ¡1 year) compared to SQL proficient users (median score of 2-3 years). The most commonly self-reported context for SQL proficient users was 'current job', while it was 'previous hobby' for SQL non-proficient users.

The Likert scale questions have a central tendency for SQL proficient users. This was not observed for SQL non-proficient users, who instead evaluated themselves between the lowest and the middle value. This might indicate that self-reporting could have been sufficient on its own, rather than having objective measures for this survey.

Objective measurements 85 out of 242 participants identified to know SQL, which had these participants answering a few extra questions. This started with three questions; two were open answers and one was multiple choice. The first question was stated as *"Can you name (some of) the keywords/syntax found in each SQL query that is used to retrieve data from a database?"*. Some faulty examples were: *"create table, data steps, etc"* and *"if then and equals does not is null"*.

These answers do not specifically (or not at all) answer the question. Hence, 66 out of 70 of the SQL proficient users answered correctly, while SQL non-proficient users had a ratio of 14 out of 15. A correct answer contains keywords like "SELECT, WHERE, FROM". These are a type of keyword typically used in almost every SQL query.

The second question relates to Set theory. As Renaud and van Biljon, 2004 alludes, knowing SQL requires the participants also to be familiar with the concepts of Set theory. Therefore, four concepts were pictured (Power set, Intersection, Difference, and Union) and questioned the participants if they could describe which concept was missing. The correct answer was Power set. Visualizing such a concept would also be more complicated than the other three. The results show that 8 out of 15 SQL non-proficient users answered correctly (Difference: 3, Intersection: 3, Union: 1), whereas 50 out of 70 SQL proficient users answered correctly (Difference: 17, Intersection: 1, Union: 2).

The third question is also manually evaluated and should contain an exact answer. The question was stated as *"SELECT Airline FROM Airlines WHERE Abbreviation = 'UAL';"*. Almost no answers were found that were incorrect, but somewhat imprecise, like *"I would expect there to be only one result as follows: 1 United Airlines UAL"*.

The only correct answer is *"United Airlines"*. This meant that SQL non-proficient users could answer correctly 9 out of 15 times, whereas SQL proficient users had a more favorable ratio of 57 out of 70.

SQL proficient users

Table 3.5 shows that out of all the participants that were identified to be SQL proficient users, the maximum number of valid queries could have been 350. Given the varying degrees of skill participants displayed, only 224 responses were submitted. However, submitted SQL queries are only usable when they can be executed. It was found that only 142 queries were executable. Out of these 142 queries, 86 were semantically equivalent to the expected query. Using Jaccard coefficient, we observe that valid participants receive an average score as low as 0.78 and as high as 1.00 per SQL Query category for Syntax evaluation. It was found to also be relatively high for the performance evaluation (average of 0.88 till an average of 1.00).

	Average Score (*n* = 70)			
	Executable (valid : submitted)	**Semantics** (correct : valid)	**Syntax** (range 0.00 - 1.00)	**Performant** (range 0.00 - 1.00)
SQL Query 1	62 : 68	55 : 62	1.00 (±0.02) valid = 68	1.00 (±0.00) valid = 62
SQL Query 2	22 : 58	14 : 22	0.83 (±0.16) valid = 58	1.00 (±0.00) valid = 22
SQL Query 3	28 : 42	9 : 28	0.85 (±0.10) valid = 42	0.98 (±0.09) valid = 28
SQL Query 4	13 : 25	4 : 13	0.78 (±0.08) valid = 25	0.90 (±0.19) valid = 13
SQL Query 5	17 : 31	4 : 17	0.91 (±0.11) valid = 31	0.88 (±0.20) valid = 17
Total	142 : 224	86 : 142	0.87 (±0.09) valid = 224	0.95 (±0.10) valid = 142

Table 3.5: Scores identified for the user group **SQL proficient user** of Study 1.

Feedback from participants

At the end of the survey, three open answer questions (OQ) were asked to allow optional feedback from the participants. These questions had the following number of valid responses:

1. **OQ1: "What should a Data Retrieval Assistant be able to do?"**: 129 responses.
2. **OQ2: "What are the most important requirements of a Data Retrieval Assistant?"**: 118 responses.
3. **OQ3: "What do you think has not been mentioned but should also be considered for a Data Retrieval Assistant?"**: 59 responses.

These responses were qualitatively thematically coded.

OQ1 A data retrieval assistant should be able to combine multiple functions into one working product. This can be observed from the wide variety of topics that participants' answers cover. For SQL non-proficient users, the most important was Accuracy (28.2%), followed by Relevancy (17.6%) and after that Quickness (9.9%). This was a shared top three of Accuracy, Recommend & Suggest, and various requirements (15.9%) for SQL proficient users.

OQ2 According to the participants, the most important function of a data retrieval assistant is quite similar, as expected. However, the top three is a bit different. For SQL non-proficient users, most important was again Accuracy (29.9%), followed by Ease of use (20.8%) and Quickness (12.5%). The SQL proficient users regarded Ease of use (24.1%) as most important, followed by Accuracy (19%) and Quickness (11.4%).

OQ3 The most common feedback from participants was about implementation details (SQL non-proficient user: 29.6%, SQL proficient user: 25%). After that, SQL non-proficient users mentioned Error tolerability (18.5%) and a shared third spot by five other categories. For SQL proficient users, this was a shared top two of (Software) limitations (16.7%) and Accessibility (16.7%).

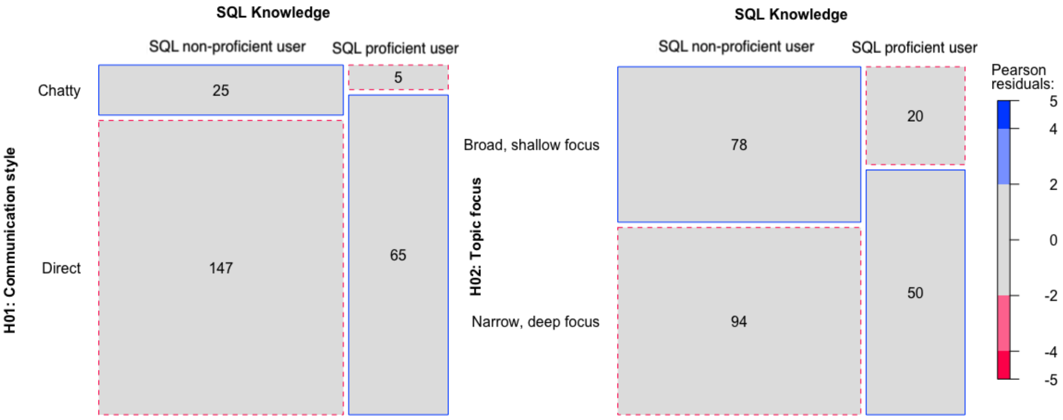
3.5.2 Hypothesis Tests

Multiple categories for hypothesis tests were defined. First, we will discuss the non-functional requirements regarding communication style and the algorithm, then the principles of Non-functional design, followed by the functional requirements. This amounts to a total of twenty hypotheses, summarized in Table 3.6, that are tested and after that corrected via a Holm-Bonferonni adjustment (Abdi, 2010). These tests are either a Chi-square test of Independence (X^2) with varying degrees of freedom or a Mann-Whitney-U test (U). A statistically significant association was found between the two variables for none of the hypotheses. Therefore, we cannot reject any null hypotheses and cannot accept any alternative hypotheses.

Non-functional requirements

Communication style Hypothesis 1, a Chi-square test of independence, was conducted between the user group and Direct vs. Chatty trade-off. Both the majority of the user groups preferred *"Direct"* (SQL non-proficient user: 85.5%, SQL proficient user: 94.4%). All expected cell frequencies were equal to or greater than five. There was not a statistically significant association between user group and communication style: Direct vs. Chatty trade-off ($X^2(1, N = 242) = 2.50$, p = .114). The association was small (Cohen, 1988), ϕ = -.102. Therefore, we cannot reject the null hypothesis and cannot accept the alternative hypothesis.

Next, for hypothesis 2, a Chi-square test of independence between the user group and topic focus was performed. Both majorities of the user groups preferred *"Narrow, deep*



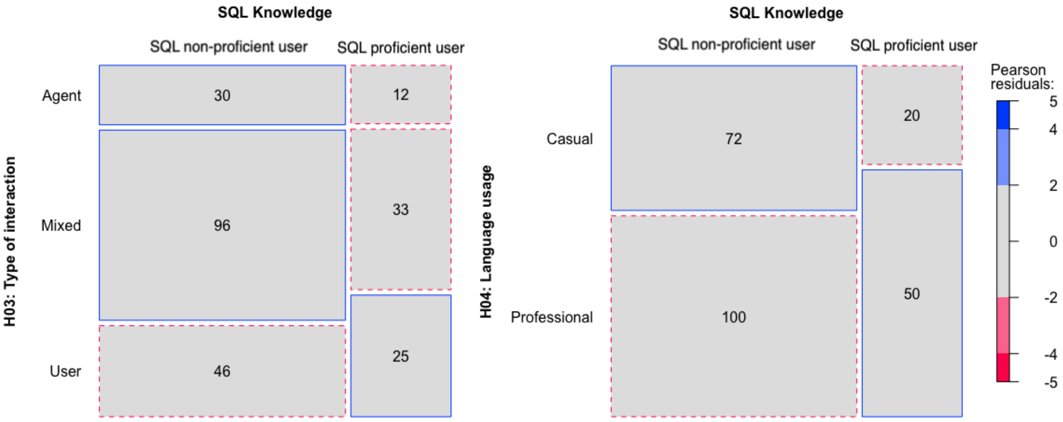
(a) H01: Direct vs. chatty tradeoff.

(b) H02: Topic focus

Figure 3.6: Mosaic plots for H01 and H02, Study 1.

focus" (SQL non-proficient user: 54.7%, SQL proficient user: 71.4%). The expected frequencies of all cells were equal to or greater than five. No statistically significant association between user group and topic focus was found. $X^2(1, N = 242) = 5.812$, p = .016. The association was small (Cohen, 1988), ϕ = -.155.

Subsequently, hypothesis 3 conducts a Chi-square test of independence between user group and type of initiative. Both the majority of the user groups preferred *"Mixed"* (SQL non-proficient user: 55.8%, SQL proficient user: 47.1%). All expected frequencies of the cells are equal to or greater than five. There was no statistically significant association between user group and type of interaction found ($X^2(2, N = 242) = 2.069$, p = .355). The association was small (Cohen, 1988), Cramer's V = .092.



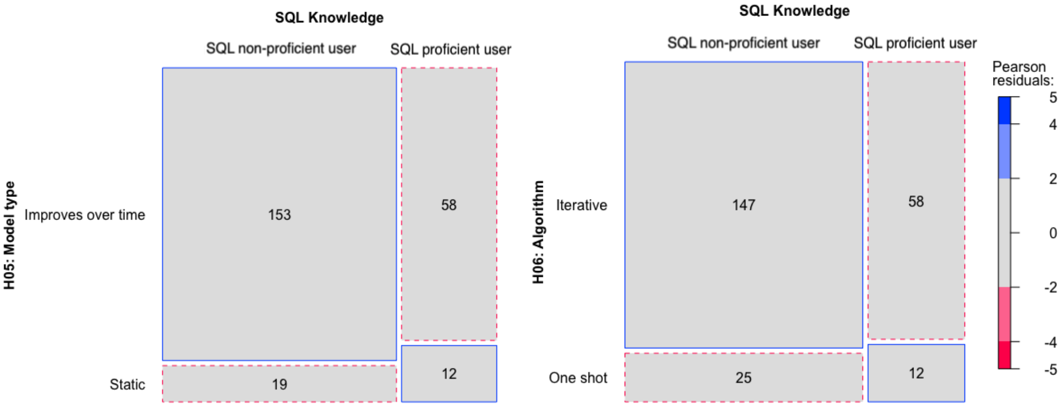
(a) H03: Type of interaction.

(b) H04: Language usage

Figure 3.7: Mosaic plots for H03 and H04, Study 1.

Then, A Chi-square test of independence between the user group and language usage was performed for Hypothesis 4. Both majorities of the user groups preferred *"Professional"* (SQL non-proficient user: 58.1%, SQL proficient user: 71.4%). The expected frequencies of all cells were equal to or greater than five. No statistically significant association between user group and language usage was found ($X^2(1, N = 242) = 3.729$, p = .053). The association was small (Cohen, 1988), $\phi = .124$.

Algorithm Hypothesis 5 was a Chi-square test of independence between the user group and model type. Both majorities of the user groups preferred *"Improves based on user feedback and becomes more effective over time"* (SQL non-proficient user: 89.0%, SQL proficient user: 82.9%). The expected frequency of all cells was equal to or greater than five. No statistically significant association between user group and model type was found ($X^2(1, N = 242) = 1.656$, p = .198). The association was small (Cohen, 1988), $\phi = -.083$.



(a) H05: Model type.

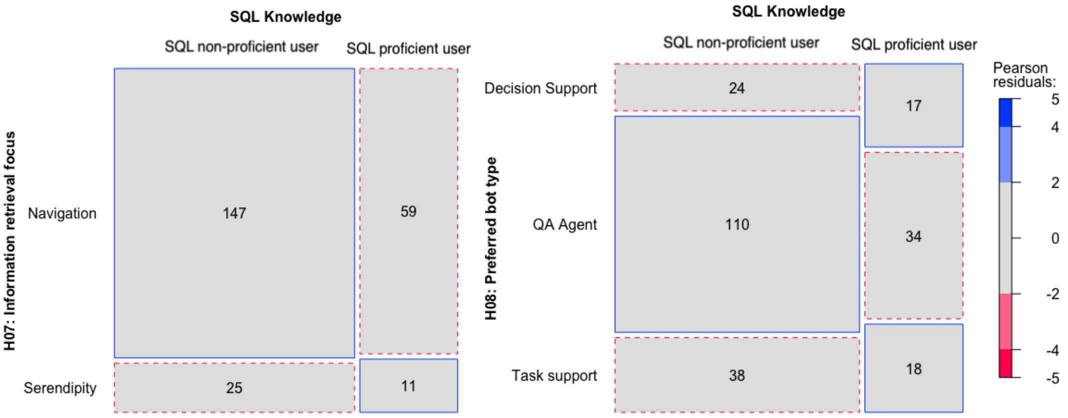
(b) H06: Type of interaction

Figure 3.8: Mosaic plots for H05 and H06, Study 1.

Then, for hypothesis 6, a Chi-square test of independence between user group and type of interaction was conducted. The majority of both user groups preferred *"Does not always have the correct answer at once, but can be adapted when it is wrong"* (SQL non-proficient user: 85.5%, SQL proficient user: 82.9%). All cells had an expected frequency greater than or equal to five. No statistically significant association between user group and type of interaction was found ($X^2(1, N = 242) = .261$, p = .609). The association was small (Cohen, 1988), $\phi = .033$.

Next, hypothesis 7, a Chi-square test of independence between user group and information retrieval focus, was performed. Both user groups had the same majority preference of *"Finding the right information"* (SQL non-proficient user: 85.5%, SQL proficient user: 84.3%). All cells had an expected frequency equal and greater than five. No statistically significant association between the user group and information retrieval focus was found ($X^2(1, N = 242) = .055$, p = .815). The association was small (Cohen, 1988), $\phi = -.015$.

Subsequently, in hypothesis 8, a Chi-square test of independence between user group and preferred bot type was carried out. The user groups shared the same majority preference of *"QA Agent"* (SQL non-proficient user: 64.0%, SQL proficient user: 48.6%). Two cells



(a) H07: Information retrieval focus.

(b) H08: Preferred bot type

Figure 3.9: Mosaic plots for H07 and H08, Study 1.

have an expected outcome of less than 5. This category *"Chatbot"* was collapsed; however, this might invalidate our result. No statistically significant association between the user group and type of interaction was found ($X^2(3, N = 242) = 7.852$, p = .049). The association was small (Cohen, 1988), Cramer's $V = .180$.

Finally, hypothesis 9, a Chi-square test of independence between user group and algorithm: accuracy vs. explainability, was conducted. The user group had the same major preference of *"Accuracy"* (SQL non-proficient user: 72.7%, SQL proficient user: 64.3%). All expected cell outcomes are five or higher. No statistically significant association between the user group and algorithm: accuracy vs. explainability, was found ($X^2(1, N = 242) = 1.675$, p = .196). The association was small (Cohen, 1988), $\phi = -.083$.

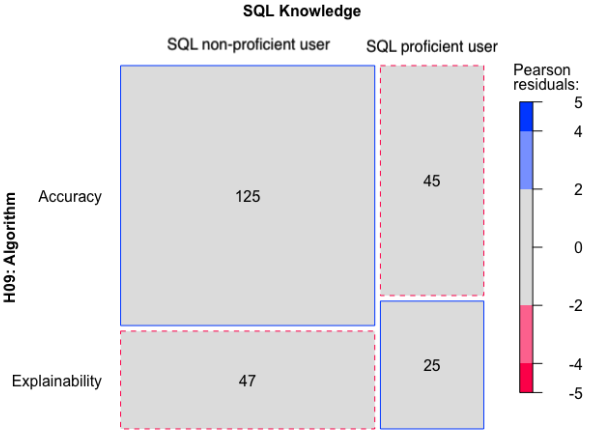


Figure 3.10: H09: Accuracy vs. Explainability tradeoff, Study 1.

Principles of Universal Design Hypothesis 10 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Low Physical Effort"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Low Physical Effort"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 126.41) and SQL proficient user(Mean Rank = 109.44), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 5176.00, z = -1.919, p = .055.

Hypothesis 11 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Equitable Use"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Equitable Use"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 127.17) and SQL proficient user(Mean Rank = 107.56), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 5044.50, z = -2.092, p = .036.

Hypothesis 12 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Flexible in Use"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Flexible in Use"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 125.17) and SQL proficient user(Mean Rank = 112.47), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 5388.00, z = -1.350, p = .177.

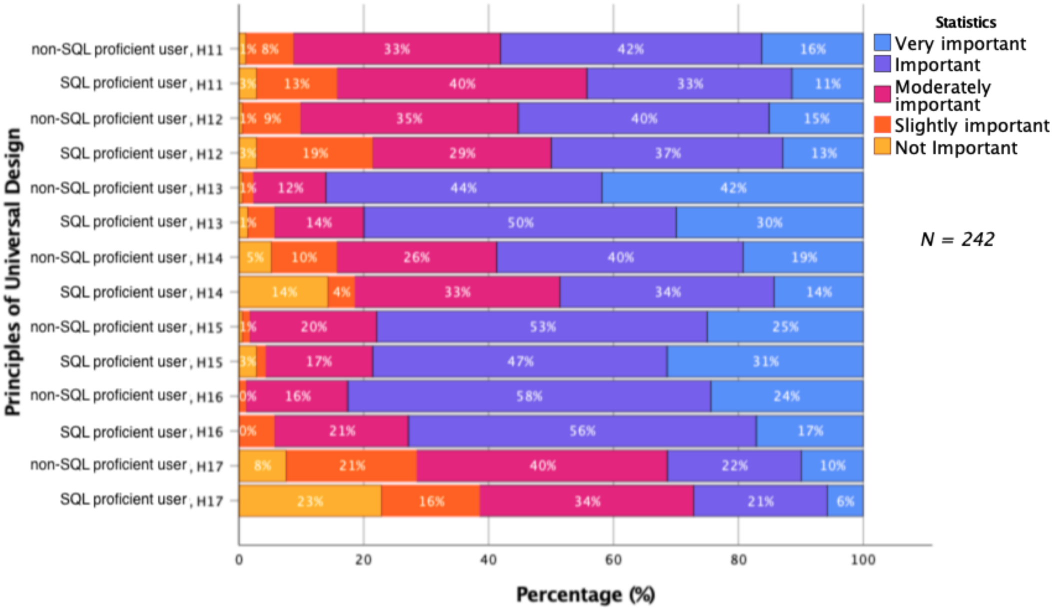


Figure 3.11: H10 - H16: The 7 Principles of Universal Design, Study 1.

Hypothesis 13 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Simple and Intuitive use"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Simple and Intuitive use"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 126.42) and SQL proficient user(Mean Rank = 109.41), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 5174.00, z = -1.864, p = .062.

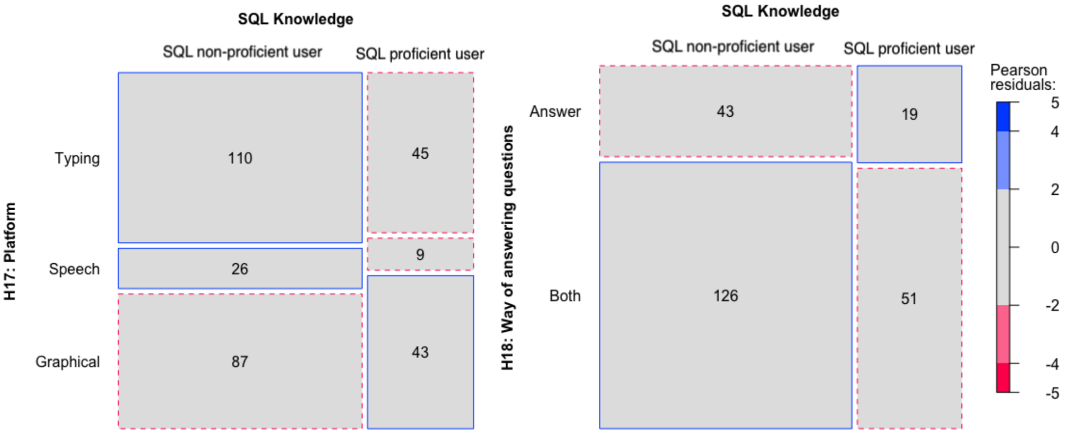
Hypothesis 14 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Perceptible Information"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Perceptible Information"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 125.68) and SQL proficient user(Mean Rank = 111.23), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 5301.00, z = -1.521, p = .128.

Hypothesis 15 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Tolerance for Error"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Tolerance for Error"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 119.78) and SQL proficient user(Mean Rank = 125.72), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 6315.50, z = 0.653, p = .513.

Hypothesis 16 a Mann Whitney U test was run to determine if there were differences in importance ranking between user groups for the value *"Size and Space for Approach and Use"*, a Principle of Universal Design. Distribution of the importance ranking were similar, as shown by Figure 3.11. *"Size and Space for Approach and Use"* was not statistically significantly different between User group: SQL non-proficient user(Mean Rank = 126.56) and SQL proficient user(Mean Rank = 109.08), $U(N_{SQL\_non\_proficient\_user} = 172, N_{SQL\_proficient\_user} = 70)$ = 5150.50, z = -1.832, p = .067.

Functional requirements

For hypothesis 17, a Chi-square test of independence between user group and interface modality was performed. The user groups' majority preference coincided with *"Typing"* (SQL non-proficient user: 49.3%, SQL proficient user: 46.4%). The expected cell outcomes are equal to or higher than five for each cell. No statistically significant association between the user group and interface modality was found ($X^2(2, N = 320) = 0.941$, p = .625). The association was small (Cohen, 1988), Cramer's V = .054.



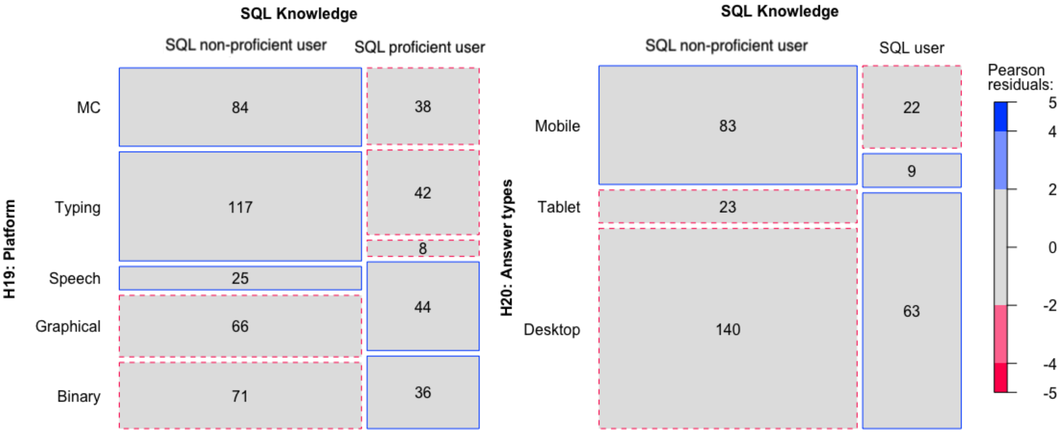
(a) H17: Interface modality.

(b) H18: Way of answering questions

Figure 3.12: Mosaic plots for H17 and H18, Study 1.

Hypothesis 18 a Chi-square test of independence between user group and way of answering questions was executed. The majority of both user groups prefer *"Both"* (SQL non-proficient user: 73.3%, SQL proficient user: 72.9%). Two cells have an expected count less than 5. This category *"Explanation"* was collapsed. However, this might invalidate our result. No statistically significant association between the user group and way of answering questions was established (X^2(2, N = 242) = 1.311, p = .519). The association was small (Cohen, 1988), Cramer's V = .074.

Next, hypothesis 19, a Chi-square test of independence between user group and platform, was conducted. The majority preference of both user groups coincides with *"Desktop"* (SQL non-proficient user: 56.9%, SQL proficient user: 67.0%). The expected cell counts are equal to or higher than five for each cell. No statistically significant association between the user group and platform was established (X^2(2, N = 340) = 3.521, p = .172). The association was small (Cohen, 1988), Cramer's V = .112.



(a) H19: Platform.

(b) H20: Answer types

Figure 3.13: Mosaic plots for H19 and H20, Study 1.

Lastly, hypothesis 20, a Chi-square test of independence between user group and answer types, was carried out. The SQL non-proficient users have a majority preference for *"typing"* (32.2%) while SQL proficient users have a majority preference for *"graphical" (26.2%)*. The expected cell counts are equal to or higher than five for each cell. No statistically significant association between the user group and platform was established (X^2(4, N = 509) = 6.609, p = .158). The association was small (Cohen, 1988), Cramer's V = .102.

#	Hypothesis	Statistic	p	α	Reject?
H01	The user group and factual vs. chatty trade-off are independent.	$X^2 = 2.5, df = 1$	0.114	0.00385	False
H02	The user group and topic focus are independent.	$X^2 = 5.81, df = 1$	0.016	0.00250	False
H03	The user group and preferred type of interaction are independent.	$X^2 = 2.07, df = 2$	0.355	0.00833	False
H04	The user group and language usage are independent.	$X^2 = 3.73, df = 1$	0.054	0.00278	False
H05	The user group and model type are independent.	$X^2 = 1.66, df = 1$	0.198	0.00714	False
H06	The user group and adaptability vs. performance trade-off are independent.	$X^2 = 0.26, df = 1$	0.609	0.01250	False
H07	The user group and information retrieval focus are independent.	$X^2 = 0.05, df = 1$	0.815	0.05000	False
H08	The user group and preferred bot type are independent.	$X^2 = 5.42, df = 2$	0.067	0.00333	False
H09	The user group and accuracy vs. explainability trade-off are independent.	$X^2 = 1.68, df = 1$	0.196	0.00625	False
H10	The user group distributions on value 'Equitable Use' are equal.	$U = 5044.5$	0.036	0.00263	False
H11	The user group distributions on value 'Flexible in Use' are equal.	$U = 5388.0$	0.177	0.00556	False
H12	The user group distributions on value 'Simple and Intuitive use' are equal.	$U = 5174.0$	0.062	0.00312	False
H13	The user group distributions on value 'Perceptible Information' are equal.	$U = 5301.0$	0.128	0.00417	False
H14	The user group distributions on value 'Tolerance for Error' are equal.	$U = 6315.5$	0.513	0.01000	False
H15	The user group distributions on value 'Low Physical Effort' are equal.	$U = 5176.0$	0.055	0.00294	False
H16	The user group distributions on value 'Size and Space for Approach and Use' are equal.	$U = 5150.5$	0.067	0.00357	False
H17	The user group and preferred interface modality are independent.	$X^2 = 0.94, df = 2$	0.625	0.01667	False
H18	The user group and preferred way of answering questions are independent.	$X^2 = 0.07, df = 1$	0.785	0.02500	False
H19	The user group and preferred platform are independent.	$X^2 = 3.52, df = 2$	0.172	0.00500	False
H20	The user group and preferred answer types are independent.	$X^2 = 6.61, df = 4$	0.158	0.00455	False

Table 3.6: A summary of the performed Chi-square tests of independence and Mann-Whitney U tests for Study 1. These were corrected using Holm-Bonferonni. None of the null hypotheses could be rejected.

3.6 Discussion

None of our hypotheses could be rejected. This means no difference between the defined SQL skill level user groups was found. In this section, we discuss some of the potential underlying reasons.

3.6.1 Sample distribution

Initially, the study was set up for a minimum sample size of 253 ($\alpha = 0.05$, Power = 0.95, Df = 8) for the Chi-square test of independence. This would allow for identifying at least three SQL skill-level user groups. However, the sample size was later adjusted to be 242 ($\alpha = 0.05$, Power = 0.95, Df = 4), given that the difference between SQL skill level user groups samples was too big. This meant that less popular preferences for some questions were below the Chi-square tests of Independence required a cell count of 5+ answers. Recruitment of more participants was not viable due to budget limitations; adjusting the question options and A priori input parameters were not preferred given that the study had run its course. Hence, the SQL skill level user groups were changed from three (Df = 8) to two (Df = 4).

The sample distribution was expected to differ regarding the SQL skill level user groups and the participants' origins. As was found in the resulting Table 3.4, most valid responses came from Prolific (SQL proficient user: 23.0%), followed by academia (SQL proficient user: 76.2%), and finally the banking industry (SQL proficient user: 50.0%).

For Prolific, SQL proficient users' ratio was higher than expected (Pilot study: 11.1%). This might be due to the different phrasing of the recruitment form. However, for other origins during the Pilot study, the participants were explicitly selected to know SQL (or not). So when running the final study, our expectancy ratio of these user groups was high: because participants from these origins are likely to have used SQL since their job often requires it.

For ING, it was expected to have recruited more participants via email. This was based on previous experiences by other AI4Fintech researchers that surveyed ING. Some estimates were 50 participants. However, the reach for the Study 1 questionnaire was estimated to be at least twice as big since more mailing lists were used. The caveat was that the Study 1 survey was expected to take participants often more than twice as long as the reference survey and that the mailing lists had to be updated. Another factor to consider is that as some banking industry participants confided, increasingly more surveys are sent out, which stifles the willingness of participants to participate. Lastly, due to the increase of emails, the day and time when the questionnaire is sent are also essential.

Eventually, 32 participants were recruited through ING, of which eleven submissions were invalid due to Microsoft Forms being limited in enforcing input validation for questions relating to ranking the importance values of *"7 Principles of Universal Design"* (Story et al., 1998). Another nine were excluded because of evaluating values either incorrectly or all equally important. This means 62.5% of the banking industry participants were excluded. One reason could be that these questions were interrelated and potentially hard to track. Another reason might be related to how the person feels about the question. A few unofficial pilot study interviews revealed that Dutch participants struggled with questions relating to ranking the importance of *“7 Principles of Universal Design”* (Story et al., 1998). The reasons could be paraphrased as participants describing themselves as a *“down to earth person”*. Therefore these values might not have been considered seriously. We are inclined to reason this might thus also be the case for some of these (Dutch) banking industry participants.

6 Out of 27 participants were regarded as invalid for academia. We think the expected total questionnaire time was the deterrent preventing more participants from completing in this category.

The limitations experienced with some of the origin groups like the banking industry, and others, lead to a lower than expected SQL proficient users sample overall. We are inclined to believe this would have impacted our eventually tested hypotheses.

There are some examples of this when we look at the limited samples of some of the origin groups. Participants from the banking industry seem to be more inclined to prefer an agent with a*“Narrow, deep focus”* (92%) compared to an overall average of 60%. Also, some of the participants from the banking industry seemed more inclined to choose *“Decision Support”* (42%) compared to an overall average of 17%. Another example is the majority preference for *“Explainable answers”* (58%) compared to the 30% overall average. These findings might indicate that perhaps origin is (more) important to consider. Reasons for this could be that the banking industry might have to abide by special regulations. For example ING, from which the participants of the banking industry were from, is a bank and thus might be required by law when employing algorithms to provide insight into how answers came to be.

3.6.2 Query language proficiency

As described in the introduction of Study 1, users of NLIDBs which translate Natural Language sentences into SQL are often identified as users lacking query language knowledge (Li et al., 2020) (Őzcan et al., 2020) (Yao et al., 2019) (Baik et al., 2019) (Zeng et al., 2020). However, no standardized assessment was found in the literature that would allow participants of Study 1 to be assessed on this proficiency. Thus an alternative approach was to determine the SQL proficiency of participants. SQL seemed most closely related because most common datasets used for NLIDBs translating natural language sentences only use SQL. Neither for SQL was there a standardized SQL assessment, but there was literature (Renaud & van Biljon, 2004) detailing the order of conceptual difficulty of SQL. This order was incorporated in Study 1 assessment of participants’ knowledge of SQL. Five queries were ordered based on this conceptual difficulty. Our sample revealed that most recruited participants could not answer at least one query correctly. Thus the SQL assessment changed into a simplified version of evaluating participants to at least be able to answer one SQL query correctly and accordingly identify as a SQL proficient user. This was thus a limitation of this study.

3.6.3 SQL Skill level assessment

Currently, a participant is assigned to be an SQL proficient user when at least one SQL query is written correctly: semantically or syntactically. Table 3.2 shows how the spread of correctly answered SQL queries is achieved. 33% Of the participants from the banking industry score higher than at least two, 8.6% of the Prolific participants accomplish this, while for academia, this is 42.9%.

Observing the newly split group of SQL proficient users into SQL beginner and SQL advanced, we observe that the SQL beginner and advanced group have conflicting views on who should take the initiative during a conversation. SQL beginner is similar to SQL non-proficient users in that they both prefer mixed-initiative (Unskilled: 56%, SQL beginner: 46%), SQL advanced prefers either mixed or user initiative at 48%. SQL advanced also prefers the agent to use Professional language (84%) compared to an overall average of 62%. These findings might suggest that better balanced with regards to SQL proficient user skill level, differentiating more thoroughly on the number of queries correctly answered, might be interesting to pursue. However, the same issues might arise as with this study since finding SQL proficient users might be difficult, especially when they are of higher skill levels, as was shown by the sample in Table 3.2, when the score of correctly answered queries increases, the sample size decrease.

3.6.4 SQL query scoring

Each query is supposed to be consecutively more difficult, as stated by Renaud and van Biljon, 2004. But there is a caveat to the SQL queries order approach, namely the assumption that SQL concept difficulty directly translates to the difficulty of Natural language. Some concepts which might be hard to grasp in SQL might be easier to understand in Natural language. Since we ask the participant to solve our question written as a Natural language query using SQL, we potentially create a bias. Another potential bias might be that, given the length of the survey, the participant might be less inclined to answer the later SQL questions than earlier, easier SQL questions. It might also affect the survey later on (**GSP**), due to long interaction might cause fatigue.

Our findings from Table 3.5 show that the fourth SQL query was least submitted, indicating it to be the hardest to answer. This could be because it contains subqueries, which might be hard to get right for participants. However, if we consider the outcome semantically, the number of correct queries follows a descending order. This can be attributed to the expectations of Renaud and van Biljon, 2004. But, considering it from the perspective of syntactic equivalence makes it challenging to provide any potential reason. The performance equivalence is based on the Spider dataset query assessment (R. Zhong et al., 2020) and numbers of nesting compared to the expected query. This also follows a descending order, which could be attributed to the increasingly more options for writing the expected query. Another reason for the average decline of queries being submitted per question can be the fatigue of participants participating. It might be that after a few SQL query answers, the participant does not want to continue creating queries. Some answers in the dataset indicated as much.

Another related issue is that there is no standardized way of assessing SQL queries in the literature. This is because SQL can be written multiple ways, meaning the same semantically, therefore most often requiring some form of manual validation. As Chu et al., 2017 described, evaluating SQL is undecidable in general. In our approach, we tackled this problem by approximating semantic equivalence for a few select SQL queries by implementing R. Zhong et al., 2020. This also inadvertently means that such an approach is not commonplace and highly specialized for this particular use case. However, the approach could be more standardized, extended, and potentially used for new applications. Currently, the limiting issue for this approach is only being able to score the query in a binary manner. It is either wrong or right. This means that the participant writing the query should be specific. Instead of writing star (), the participant should specifically state the relevant columns required. In practice, this specificity might not be that important; however, it would be for the currently used approach. Also, since the approach is approximate, it means that when evaluating, it might not always provide the right answer. According to the paper that introduced this approach (R. Zhong et al., 2020), $\geq 99\%$ of the neighboring queries are identified for the dataset used.

3.6.5 Syntax support

Currently, SQL syntax support for queries only aims to provide ways to query databases, the Data Manipulation Language (DML) of SQL. There is no approach attempted to offer other possibilities, like modifying or creating entries for the database or creating the database schema. Another problem is the limited support of supported syntax. For example, the Spider dataset covers no full, left, right, cross, and inner joins.

3.6.6 Assumption and limitations of NLIDBs

Some requirements might make it hard for participants to use NLIDBs that are not already familiar with some concepts. Participants need to be familiar with concepts like knowing what tables and columns are what column names mean. Another limitation to current NLIDBs is that they need to have normalized databases, such that there are no circular references or paths since this would only make correct predictions harder.

3.6.7 Data cleaning

We expect a sizable amount of participants who self-identified as SQL proficient users to have failed to answer some questions correctly due to fatigue. These SQL proficient users often spent more time finishing the questionnaire than SQL non-proficient users. In such a case, we talk about **GSP** (Gadiraju et al., 2015). These participants have no ill intentions but might eventually fail to answer correctly. Results reveal that 34.1% of the self-reported SQL proficient users were removed during data processing, compared to SQL non-proficient users that had a percentage of 47.4%. So, while SQL non-proficient users were more likely to be removed during data processing, we believe this category is more likely to contain malicious participants.

3.6.8 Expertise criteria

The criteria used to identify participants was limiting because it can produce false positives and negatives. One such false positive is that the participant cheats and inserts a random SQL query, recognized but not valid. Some false negatives are when participants use a dialect different from SQLite or make a typo that causes the query not to be identified.

The implementation somewhat alleviates both these false negatives. An example is the implemented grammar matching algorithm and the automatic refactoring of some of the syntax of other SQL dialects found in some SQL queries. This, however, did not always succeed in alleviating the problem. The False positives are alleviated by employing multiple mechanisms to evaluate a query (Query is executable, semantically equivalent, syntactically equivalent, performance-wise equivalent). This means all the relevant attributes should be named for it to be recognized as syntactically correct. However, false positives that were accepted but were wrong have not been found when manually evaluating the dataset.

Chapter 4

Study 2

Section 3.5 of Study 1 revealed that we cannot conclude that there is a difference between the preferred requirements of the segmented groups. However, a later Section 3.6.1 reveals exploratory findings, which indicate that participants originating from the banking industry have a preference for explainability over accuracy, which is different from participants from other origins or the segmentation as defined in Study 1. While the sample size is limited (*N* = 12), the outcome is likely. The banking industry is bound by many laws and regulations, requiring that algorithms provide insight into how answers are created.

Thus inspired by these findings of explainability preference and a user study from Narechania et al., 2021 which claimed that assessing the correctness of answers provided by an NLIDB model "[...] can be challenging for people who lack expertise in query languages", Study 2 was created.

In Study 2, we want to quantify a color-coding technique, inspired by Narechania et al., 2021, to help assess users' correctness of answers provided by an NLIDB model. This color-coding technique is enabled by modifying an existing model (Wang et al., 2020) to expose the underlying relations it uses to go from input to output. To link a natural English sentence to an SQL query, certain words from the sentence are implicitly linked to parts of the SQL query. This SQL query will refer to specific columns in the database. So, we hypothesize that by revealing these relations, users should be better able to trace, verify and explain answers generated by an NLIDB model.

Helping users quantify the correctness of NLIDB models is important because the latest NLIDBs often fail to provide the correct answer. NatSQL (Gan et al., 2021) was at the time of writing number two on the Spider leaderboard, a commonly used dataset (Yu et al., 2018) for NLIDBs. The model has an average accuracy of 73.3%. When the complexity of the generated SQL increases, the accuracy can drop to an average of 51.8% for queries from the Spider dataset (Yu et al., 2018). Consequently, at least a quarter of the queries generated by the NLIDB model might fail.

So in Study 1, preferences for an NLIDB were investigated from the user's perspective. This user either knew how to use SQL (SQL proficient user) or not (SQL non-proficient user). These preferences were commonly found in literature and related implementations. Study 2 uses these preferences to create an application, which we call **IRA** (Information Retrieval Assistant).

The implementation of Study 2 is thus aimed at users who lack the expertise in query languages and want to be able to assess the correctness of the model. The requirements are shown in Table 4.1. The way Color-coding is used in the application is a form of local self-explanation to help assess the correctness of queries (Danilevsky et al., 2020). The technique used to enable Color-coding is called **Attention** and is a technique commonly used in deep learning for NLP.

The participant is provided with the original information of the database, the natural language sentence question the NLIDB model tries to answer, and if Color-coding is enabled, show which columns are relevant for answering the question (Figure 4.4). Color-coding is enabled for 50% of the participants.

This chapter contains the experimental setup, implementation, variables, statistical hypothesis testing, data preparation, results, and discussion.

#	Requirements choice	Implemented majority preference?
R01	Direct or Chatty: Direct (88%, 2 options)	**Yes**.
R02	Narrow, Deep focus or Broad, Shallow focus: Narrow, Deep focus (60%, 2 options)	**Yes**, only one database focus.
R03	User, Agent, Mixed initiative: Mixed initiative (53%, 3 options)	**Yes**, uses an interactive interface, where we nudge the participant through the correct flow.
R04	Casual or Professional: Professional language (62%, 2 options)	**Yes**.
R05	Self-improving or Fixed: Static (13%, 2 options)	No, due to nature of the study.
R06	One-shot or iterative: One-shot (15%, 2 options)	No, due to nature of the study.
R07	Serendipity or Navigation focus: Navigation (85%, 2 options)	**Yes**, aimed for navigation of data.
R08	Type of Agent: QA Agent (60%, 4 options)	**Yes**, both Task support and QA Agent.
R09	Accuracy ot Explainability: Explainability (30%, 2 options)	No, due to nature of the study.
R10	Low Physical Effort: Important, M = 3.996, SD = 0.709	Attention: **Yes**, Baseline: No.
R11	Equitable Use: Important, M = 3.566, SD = 0.910, Important	**Yes**, taken colorblindness into account.
R12	Tolerance for Error: Important, M = 4.012, SD = 0.791	No, participants were unable to change answer after submission.
R13	Flexible in Use: Important, M = 3.537, SD = 0.925	No, due to nature of the study.
R14	Perceptible Information: Moderately important, M = 3.492, SD = 1.120	**Yes**, customizable pagination, buttons with distinct colors.
R15	Simple and Intuitive use: Important, M = 4.186, SD = 0.806	**Yes**, modern user-friendly UI (Material Design[1])
R16	Size and Space for Approach and Use: Moderately important, M = 2.995, SD = 1.113	No, due to nature of the study.
R17	Interface modality: Typing (48%, 3 options)	**Yes**.
R18	Importance answer or explanation: Both (73%, 3 options)	**Yes**.
R19	Platform: Desktop (60%, 3 options)	**Yes**.
R20	Answer types: Typing (30%, 5 options)	**Yes**.

Table 4.1: Implemented requirements based on Study 1, with preference percentage observed for Study 1, total number of preference options and reasoning for implementing majority preference (or not), Study 2.

4.1 Experimental Setup

4.1.1 Objective

As stated in the introduction, users unfamiliar with query languages like SQL might have issues connecting their inquiry to the result provided by the model. Such a result is often presented as a table with columns and entries. Some of these properties might be unfamiliar to the user, especially if the results provided by the model are wrong.

Therefore by using Color-coding, a local self-explanation technique using attention (Danilevsky et al., 2020), columns and (some) words in the inquiry of the participants are given the same color such that participants can visually see the relation. Another factor taken into account is the SQL difficulty as defined by the Spider dataset (Yu et al., 2018). This dataset categorizes queries into four difficulty levels: Easy, Medium, Hard, and Extra-hard. For this setup, we investigate the performance difference by assessing the explanation technique and SQL difficulty difference:

RQ3 How does Color-coding influence the performance per query?
RQ4 How does the combination of Color-coding and the SQL category influence the performance per query?
RQ5 How does the SQL category influence the performance per query?

4.1.2 Case study

To investigate the previously posed research questions, participants unfamiliar with SQL are recruited. They are tasked to identify if the IRA model can correctly link a given question to the correct data. These participants are randomly assigned and evenly distributed between the condition Color-coding and Baseline. Study 1 showed that Prolific participants were least likely to be skilled in SQL (Prolific: 23%, Banking Industry: 50%, Academia: 76.2%) and so the recruitment is limited to the Prolific platform.

Procedure

The survey has five parts as visually represented by Figure 4.1.

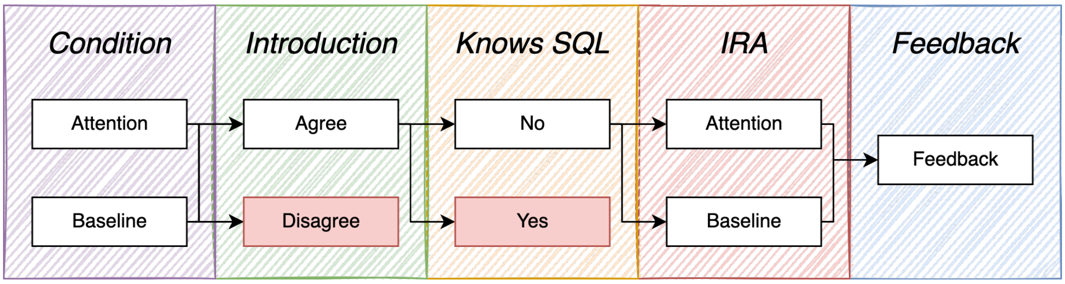


Figure 4.1: The survey flow for each participant of Study 2.

Condition. Every participant is assigned the condition at random, which means 50% for each condition to create a balanced sample.

Introduction. The introduction summarizes the survey and offers general guidelines. This is to prepare the participant. The participant can decline, which ends the survey or accept to continue. Next, participants are asked to conceptually visualize a large collection with multiple categories. Here, we hope to gain insight into whether participants relate this to a digital, analog (or combination) concept.

Knows SQL. Each participant who answers to be familiar with SQL is excluded from the survey. This is because unfamiliarity with SQL is a requirement for participating in the survey, as was stated in the introduction and recruitment information.

IRA. The application conditionally colors their columns and parts of the inquiry that are related. This is only enabled for participants assigned the condition Color-coding.

Feedback. By using the 7 Principles of Universal Design (Story et al., 1998), System Usability Scale (Brooke, 1996) and a feedback question, participants, are given the opportunity to provide feedback on their experience with IRA. The open question responses were qualitatively coded.

Participants Recruitment

The quality of the data is ensured by enforcing multiple measures. In Figure 4.2 all measures that were taken for Study 2 are mentioned.

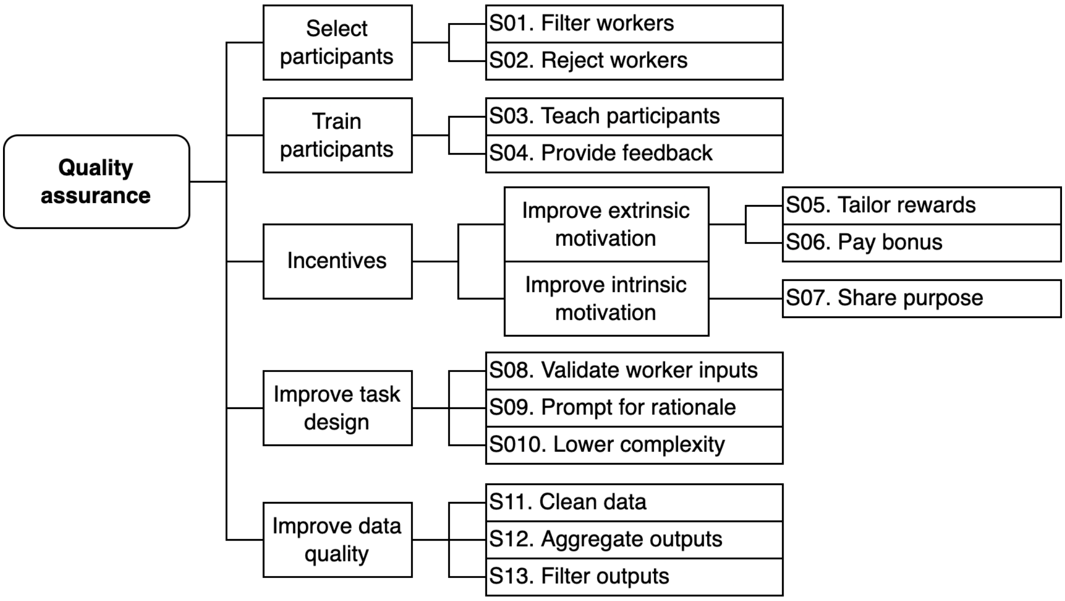


Figure 4.2: Quality assurance strategies and their corresponding measures taken for Study 2 (adapted from Daniel et al., 2018).

We used the platform Prolific to recruit crowd-workers. These workers were filtered (S01) according to the following criteria:

(1) **Minimum age of 18.** Legal limit by Prolific.
(2) **Fluent in English.** The survey is completely in English: participants need to know English.
(3) **First Language is English.** Study 2 is a technical topic, which adds a layer of complexity for the participants. Thus to avoid adding another layer of complexity, translation, we assume native English speakers will have less difficulty than Fluent speakers.
(4) **Use desktop.** The website is not optimized for small screens because the general NLIDB use-case is desktop-focused. NLIDB applications often display (large) tables, which preferably work best on large (wide) screens.
(5) **Approval rating of 95 - 100.** Higher approval rating might indicate higher quality participants.
(6) **Minimum previous submissions of 30.** Combining this with the earlier measure ensures more advanced Prolific participants are recruited, rather than participants with a high approval rating because they completed a few tasks correctly.

The criteria S02 and S11-S13 are described in Section 4.5.

IRA includes a small tutorial to train participants(S03), explaining how it works. It uses an interactive chat window for that. Via this interactive chat, a new message is displayed every few seconds. The display of these messages is timed according to the upper range of the average reading speed found by Brysbaert, 2019. Whenever the participant provides an answer that is not how the survey had intended, the survey offers feedback on improving the response (S04).

Moreover, the Prolific platform tries to extrinsically motivate participants by compensating them according to the time put in (S05). The hourly rate of £7.50 per hour was used for the study. Participants that experienced difficulty or were found to have put in more effort were compensated through a bonus (S06). This was equal to the estimated time spent and hourly rate. Also, the purpose of Study 2 was shared with the participants in the recruitment message on Prolific and introduction of the survey S07). These three incentives were put in place to increase the data quality.

Next, to improve task design, the survey enforced input for the required questions, checked for incorrect answers, and ensured a minimum number of words for questions requiring a rationale (S08). Accordingly, most questions need a justification from the participant to provide higher quality data (S09). Also, jargon is replaced with more accessible terms and concepts to decrease the complexity of the task at hand (S10). One such example is replacing *"NLIDB"* with *"assistant that helps to retrieve information"*.

Other measures taken that were not mentioned by Daniel et al., 2018 are the following:

- Participants can provide optional feedback.
- Question answer order is randomized.
- Enforcement of time constraints, which exclude too fast participants.
- Attention checks to ensure the participant is paying attention.

These measures were tested through a Pilot Study.

Pilot Study Twenty-one participants were recruited, of which seven were deemed invalid. This was due to intended (failing attention checks) and unintended measures (technical difficulties). Examples of unintended technical issues were the limitation of the webserver to deal with single-page websites (SPA), infinite loops, UI inconsistencies, and no presence of a timer. These issues were fixed for the final study and explained in Section 4.5.

4.2 Implementation

IRA is set up to identify whether the condition Color-coding performs better than Baseline. For this, we make use of the queries found in the Spider dev dataset (Yu et al., 2018). However, this dataset contains 1988 queries of varying levels of SQL difficulty. Therefore, we need a subset of the available queries since not all can be displayed.

4.2.1 Query selection

The study setup is limited by a finite amount of time due to participants' interaction fatigue. We cover each type of SQL difficulty at least once, but mostly twice. There are four categories (Easy, Medium, Hard, and Extra-hard). There are eight queries displayed to each participant. One of these queries is a tutorial. A potential learning bias is prevented by randomly shuffling the order at which eight queries are shown.

This selection is also limited by some of the requirements found in Study 1. The participants from that study preferred an NLIDB with *"Narrow, Deep Focus"* (60%). Therefore we limit the databases used for IRA to one, preventing context switches that might have confused some participants.

Limitations to the screen size of participants are also taken into account. Database tables often contain more records than can be displayed on one screen. This makes verifying answers harder. Databases that at most have only 20 rows per table are included.

Next, the used RAT-SQL (Wang et al., 2020) model has a limited accuracy. Therefore, queries that cannot be predicted accurately are excluded.

The values stored in the leftover databases are checked for errors. No errors were found, but unlikely scenarios were discovered and excluded. For example, the database *"employee_hire_evaluation"* contained information of stores owned in Finland while the employees were registered to be living in England. This is an unlikely scenario and for that reason excluded.

Moreover, Google Trends[2] is used to determine which topic has the highest interest. The most popular word is *dog* and hence the **dog_kennels** database is selected.

Lastly, we ask a group of three people fluent in English, with different backgrounds (Person 1: Knows no SQL, does not use Excel, Person 2: Knows no SQL, uses Excel, Person 3: Knows SQL, uses Excel), to rate which NL query they think are least ambiguous, most natural-sounding queries.

4.2.2 Architecture

In Figure 4.3 the architecture of IRA is displayed. Every box colored blue is ran on a virtual private server (VPS) using Docker[3], while data is stored using Firebase[4]. These colored boxes each represent a micro-service performing only one task. The Redis[5] database

[2] https://trends.google.com/

[3] https://www.docker.com/

[4] https://firebase.google.com/

[5] https://redis.io/

contains information regarding the schemas of each database. Consequently, it returns the available databases, the tables in a database, the columns in a table, and the synonyms used for each of these components. The Spider dataset (Yu et al., 2018) translates the component acronyms to expanded form such that it is easier for a human and a neural model to understand the concept it represents.

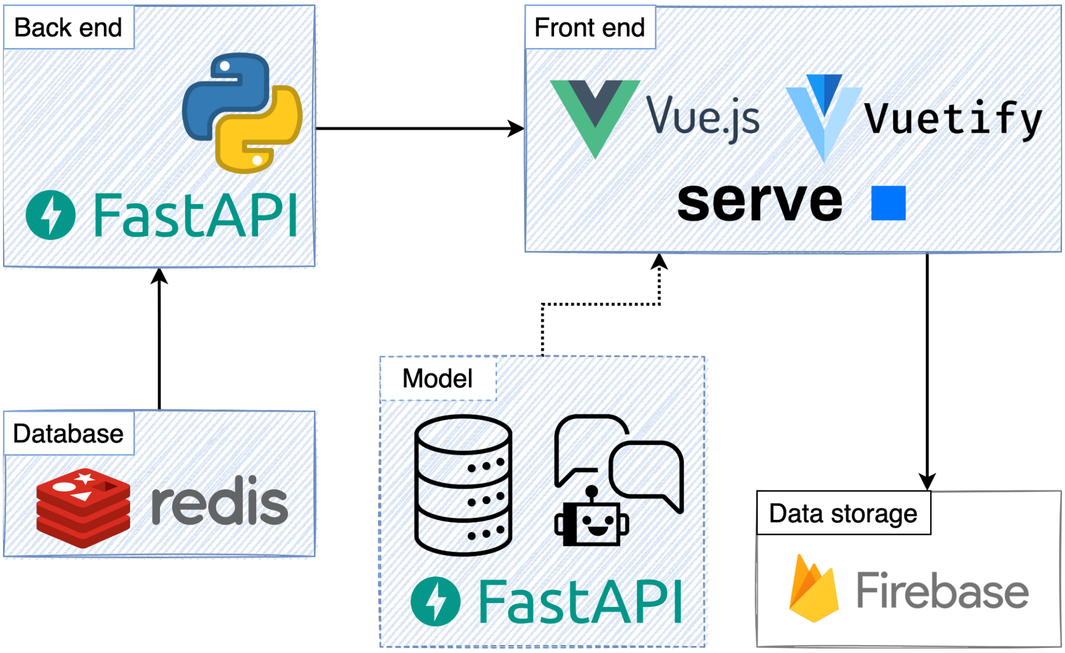


Figure 4.3: Architecture of IRA, Study 2.

The back-end contains the databases and, combined with the Redis database, returns the data contained in the database with their respective expanded formats to the front-end. This is ran using a Python[6] REST-API called FastAPI[7].

A live model has been implemented which could let the participant interact with the NLIDB model RAT-SQL (Wang et al., 2020), a neural model integrated into a FastAPI implementation. However, interacting with such a model proved to be unstable. This meant it could shut down after a random number of interactions (related to memory issues), resolving participant queries at varying amounts of time. To limit the variability of the participant study, it was opted to query the output of the model offline and integrate it within the front-end.

The front-end connects to the back-end, model, and data storage. It uses VueJS[8] combined with Vuetify[9] to create a single page application (SPA). This is a client-based application with front-end constraints and little back-end constraints, which could allow a mali-

[6] https://www.python.org/

[7] https://fastapi.tiangolo.com/

[8] https://vuejs.org/

[9] https://vuetifyjs.com/en/

cious participant to modify the data. However, no malicious attempts were administered for the Pilot and Final study.

4.2.3 User Interface

The user interface has a sidebar, shown by ⑦, which displays the stage the participant is in. Every page, except as shown by Figure 4.4, contains forms. These forms enforce answers to be given in a certain format and require to be answered to proceed with the questionnaire. Figure 4.4 displays the interaction of a participant with IRA and Color-coding enabled. IRA is a chatbot that guides the participant through the process of verifying if the Question posed, shown by ①, leads to the correct answer, shown by ⑤. A timer is displayed, shown by ②, to make the participant aware of the time spent. ③ Shows which part of the question the model uses and links it to the relevant table. In this case, at ④ and ⑥ it is connected to the table "Dogs" and column *"Age"*. At ⑧ the most recent messages are shown. In this case, IRA instructs the participant to start the tutorial by clicking the green button (⑩). ⑨ Shows that answering is disabled for the participant at the moment. Not all tables fit the screen vertically. Pagination of the table is enabled (⑫), which allows the participant to select how many rows should be displayed and to select the page. If the screen size does not fit horizontally (⑪), the participant can scroll sideways.

Four states are used for ⑩, each color indicates a different phase. Green is used to start the tutorial as noted in the chat (⑧). Red is used to stop the tutorial and finish the application, while blue is used when a participant wants to submit an answer (⑨). Loading is used when IRA has sent a message and processed the participants' request. This feedback mechanism provides participants an indication of when they have to wait.

There are two interfaces, dependent on the condition assigned to the participant. This is shown in how the Question (①) and table columns are displayed. This is exemplified by the difference between Figure 4.4 and Figure 4.6.

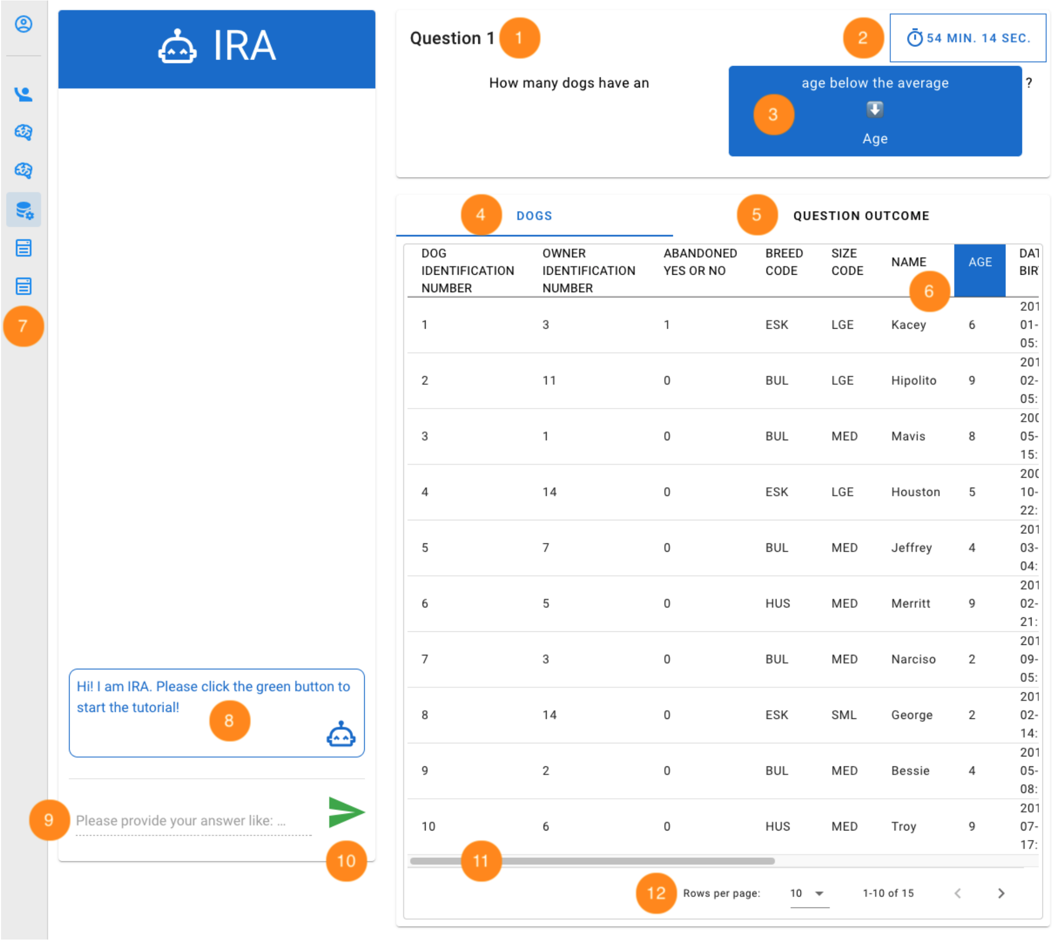
4.3 Variables

The independent, dependent, subject, controlled, and uncontrolled variables are described below.

4.3.1 Independent Variables

The independent variables are used to assess our research questions.

- **Condition.** Used to answer **RQ03** and **RQ05**. In NLP, it is common to use some form of explainability techniques like Attention (Danilevsky et al., 2020). In this paper, Danilevsky describes four categories, of which, in our case, only the Local Self explanation is relevant. This means explaining a single prediction using the



IRA

Question 1

How many dogs have an age below the average ?

Age

54 MIN. 14 SEC.

DOGS

QUESTION OUTCOME

DOG IDENTIFICATION NUMBER	OWNER IDENTIFICATION NUMBER	ABANDONED YES OR NO	BREED CODE	SIZE CODE	NAME	AGE	DAT BIR
1	3	1	ESK	LGE	Kacey	6	201 01- 05:
2	11	0	BUL	LGE	Hipolito	9	201 02- 05:
3	1	0	BUL	MED	Mavis	8	200 05- 15:
4	14	0	ESK	LGE	Houston	5	200 10- 22:
5	7	0	BUL	MED	Jeffrey	4	201 03- 04:
6	5	0	HUS	MED	Merritt	9	201 02- 21:
7	3	0	BUL	MED	Narciso	2	201 09- 05:
8	14	0	ESK	SML	George	2	201 02- 14:
9	2	0	BUL	MED	Bessie	4	201 05- 08:
10	6	0	HUS	MED	Troy	9	201 07- 17:

Rows per page: 10

1-10 of 15

Hi! I am IRA. Please click the green button to start the tutorial!

Please provide your answer like: ...

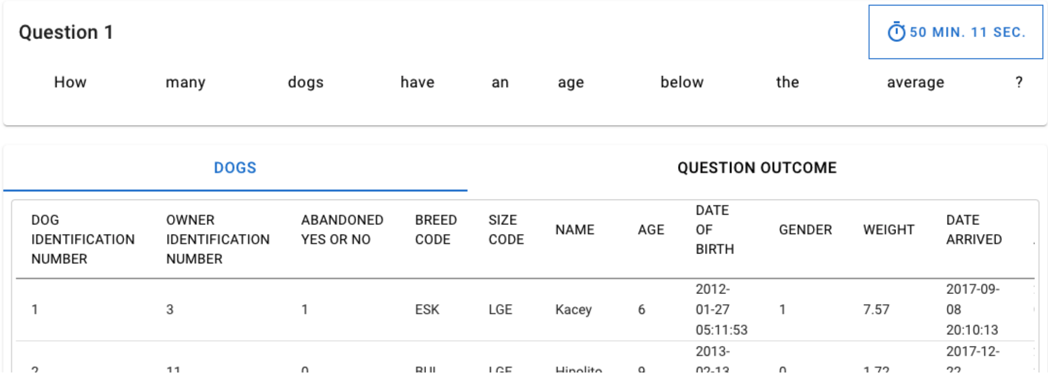
Figure 4.4: Condition Color-coding enabled for IRA, Study 2.

Figure 4.5: Button states (Continue, Stop, Start, Loading) for IRA, Study 2.

model itself. So in our case, we use the technique called attention and use it to apply color-coding to visually color important words (feature importance). This means some words of the query and columns are given the same color to show the relation. This was also employed by the DIY implementation (Narechania et al., 2021) in a different study setup.

- **SQL Category.** Used to answer **RQ04** and **RQ05**. The Spider dev dataset (Yu et al., 2018), categorizes SQL queries into four categories: Easy, Medium, Hard, and Extra-hard. This definition is described in their GitHub project[10] and assigns difficulty according to how many SQL components each query has. Some components are

[10] https://github.com/taoyds/spider/tree/master/evaluation_examples



Question 1

How many dogs have an age below the average ?

DOG IDENTIFICATION NUMBER	OWNER IDENTIFICATION NUMBER	ABANDONED YES OR NO	BREED CODE	SIZE CODE	NAME	AGE	DATE OF BIRTH	GENDER	WEIGHT	DATE ARRIVED
1	3	1	ESK	LGE	Kacey	6	2012-01-27 05:11:53	1	7.57	2017-09-08 20:10:13
2	11	0	BUL	LGE	Hipolito	9	2013-02-13	0	1.72	2017-12-22

Figure 4.6: Condition Baseline enabled for IRA, Study 2.

identified to be harder than others, which also increases the difficulty.

This allows us to test 2 (Conditions) x 4 (SQL categories) = 8 conditions to be tested in this setup.

4.3.2 Dependent Variables

We measure performance metrics required for each research question:

- **Time spent in seconds.** This is used to answer **RQ03**, **RQ04**, **RQ05**. This metric allows measuring how long each participant interacts with a question.
- **Correctly answered.** This is used to answer **RQ03**, **RQ04**, **RQ05**. Each participant answers each query with an open answer that is enforced to start with either yes or no and is followed by reasoning. This answer is evaluated according to what is described in Section 4.5.2. There we make a distinction between incorrect and partially correct. However, for this variable, we consider incorrect and partially correct to both be counted as a zero score and therefore incorrect.

4.3.3 Subject Variables

The first open question posed in the questionnaire (OQ4) investigates which concepts participants think of when visualizing a large collection of information with categories. Participants might link this to digital, analog, or a concept shared by these two categories. The participant's familiarity with these concepts might be a proxy for their performance.

4.3.4 Control Variables

The study setup enforces randomization but also some form of standardization:

- **Tutorial.** The first question, which is randomly selected for the questionnaire, is a tutorial.

- **Expected answer.** The setup enforces at random if it expects the answer for the randomly assigned query to be either true or false.
- **Color blindness.** When a participant is assigned Color-coding, the colors used are either a shade of blue, red, or orange, based on the IBM color blind safe color palette. This palette seemed most distinctive, as shown by (Nichols, n.d.).

4.3.5 Uncontrolled Variables

The following variables are collected but were not controlled for during data gathering:

- **Expected answer "Yes" or "No".** The participant is assigned a query that is either wrong or right. Options: yes, no.
- **Number of color-coded columns.** There can be up to three color-coded columns for each question. This is dependent on which queries were selected for the questionnaire as described in Section 4.2.1. Options: #1, #2 or #3.
- **Type of error.** There are four types of errors identified by the RAT-SQL paper (Wang et al., 2020) in the ablation analysis. Since there were eight queries as described in Section 4.2.1, we assign each error to two queries, allowing us to create 16 queries in total. Options: Adding column, wrong column, missing column, missing where clause.
- **Query type.** Each query can be identified by a different type of operation performed. For example, selecting an attribute, counting attributes, and other aggregation possibilities. Options: Count, Average, Select.

4.3.6 Potential Confounding Variables

More variables were measured during the questionnaire, which might have the potential to be confounding:

- **Survey: Time spent in seconds.**
- **IRA: Time spent in seconds.**
- **Correctly answered total.** Based on the number of correctly answered queries (0: no queries correct, 7: all queries correct).
- **Expectancies of an NLIDB regarding the 7 Principles of Universal Design.** In Study 1, these 7 Principles of Universal Design (Story et al., 1998) were measured to guide design decisions for Study 2. Now this question is used to compare. Options: Not important, Slightly important, Moderately Important, Important, Very important.
- **Embodiment of values for IRA regarding the 7 Principles of Universal Design.** We use these values to compare Study 1 with Study 2.
- **System Usability Scale (SUS) per question.** SUS has been used to measure the perceived usability of a system (Brooke, 1996). In related research (**narechania˙diy˙2021**), it has been used as a quick tool of measurement. Here we use it that same way.

4.4 Statistical Hypothesis Testing

For this study, we explored six hypotheses between a mix of independent variables (Color-coding vs. Baseline), (SQL categories: Easy, Medium, Hard, Extra-hard) and dependent performance variables (time) and (score). Since the data is parametric we use ANOVA. G*Power (Faul et al., 2007) was used to calculate the required sample size. The Apriori parameters were set to an Effect size (f) of 0.25 (medium according to Cohen (Story et al., 1998)), α of 0.05 to correct for Type-I error inflation, Power (1 - β) of 0.95 to correct for Type-II errors and the number of groups to 8. This amounts to a total sample size of 360. However, due to financial constraints, this sample size was not met. A sensitivity analysis revealed an effect size of 0.281, given that the final sample size is 284. The family-wise error rate will be corrected via the Holm-Bonferroni method. Tukey HSD test is performed for statistically significant ANOVA results for further analysis. This type of test already automatically corrects for a family-wise error rate.

	H21, H22	**H23, H24**	**H25, H26**
Statistical test	One-way ANOVA	Two-way ANOVA	One-way ANOVA
Independent variable(s)	(Color-coding vs. Baseline)	(Color-coding vs. Baseline) +(SQL categories)	(SQL categories)
Dependent variable(s)	Time, Score		

Table 4.2: Types of statistical tests for Study 2.

4.5 Data Preparation

4.5.1 Data Cleaning

Data control measures were taken to ensure the quality of the data (Figure 4.2 S02 and S11 - S13). However, given that the user group is now only limited to Prolific participants, it simplifies the process. The five malicious categories as identified by Gadiraju et al., 2015, was again used to classify participants' behavior as was found in the dataset:

- **Rejected submissions.** 12 Submissions were rejected on Prolific based on criteria like failed attention check, being a low effort response, or missing a completion code. These participants were identified to be **RB** often and sometimes **GSP**.
- **Timed-out submissions.** 9 Submissions timed out on Prolific. This can be due to many reasons. The time limit set by Prolific was at 56 minutes for completion. For these submissions, we found no malicious intent.
- **Low effort submissions.** The following categories were identified:
  - **Incomplete submissions.** 133 Submissions were excluded because only the introduction had been completed. This either indicated that they failed an attention check (**RB**, **GSP** or **FD**), decided not to participate after reading the introduction, were familiar with SQL (**IE**) or thought themselves to be ineligible for other reasons (**IE**).
  - **SQL submissions.** 2 Submissions contained references to SQL. This indicated familiarity with SQL and thus were excluded (**IE**).
  - **Unclear submissions.** 17 Submissions were excluded since they either contained unintelligble or irrelevant answers to questionnaire (**RB** or **FD**).
- **Technical issue submissions.** The prototype suffered the following issues:
  - **Timestamp error.** 5 Submissions contained errors in the timestamp, creating extreme outliers. This can be attributed to the implementation of IRA and the participants' browsers. This specific method for creating timestamps unintentionally works differently per browser, platform, and version.
  - **Time limit exceeded.** 1 Submission exceeded the time limit set by Prolific and was consequently excluded. The Prolific platform determines this time limit.
  - **Database results error.** 15 Submissions showed that the participant was unable to retrieve all the relevant data from the database automatically.

In total, 173 submissions were excluded from the dataset. The total number of measured submissions is 476, of which 303 are valid. None of the submissions were manipulated.

The submissions were semi-automatically verified. This meant that participants that failed attention checks were automatically flagged, while participants that mentioned SQL for open answers were first identified by manual verification. Then all 1988 queries were manually annotated, which is elaborated in Section 4.5.2.

The last question of the questionnaire was an open answer type question. This allowed participants to provide feedback about their user experience. Only 88 out of the 303 participants provided feedback.

4.5.2 Data Preprocessing

The dataset is divided into three sections: Pre-survey, IRA, and post-survey. These three separate parts are merged and processed to create a dataset in which each entry is one unique participant response.

Merging Datasets

- **Conversion.** All three sections are converted from JSON format to CSV.
- **User evaluation.** Every participant that completed the survey was asked to provide their judgment in section Application. Their judgment was automatically evaluated by extracting the first word of each judgment, which was enforced to be either ”yes” or ”no”.
- **Time spent.** The timestamp is created locally for each participant. These are used to time their interaction time, the time it took to complete the survey, and the time it took to complete the IRA part. These timestamps all had to be converted to seconds.
- **Privacy.** The Prolific id is replaced by a random number. No other personal information was gathered from the participant.

Query Annotation

An automated approach to evaluate queries overestimates participants' ability to correctly identify when an answer provided by the IRA model is wrong or right. This affects the score performance. Automatic evaluation finds an average score of 70% correct, while manual evaluation leads to a score of 35% correct.

Two processes are performed as shown in Figure A.1 of Appendix A. The first process evaluates whether the participant correctly affirms the expected answer by starting their response with either “yes” or “no”, followed by their reason. If the participant starts with the correct first word, followed by a correct reasoning, the query is correct. If one part is correct, it is partially correct, while it is called incorrect if both parts are wrong. However, with regards to the score, partially correct and wrong are both identified as zero scores. In Appendix A the process is elaborated.

The second process evaluates which annotation category the query falls into. A correctly answered query can only fall into the category “Nothing”. In contrast, every category is relevant for partially correct and incorrect queries. All of the category “Technical error” and some of the category “Anomaly” are excluded as mentioned in the Section 4.5.1.

These processes were affirmed by making use of inter-rater reliability scores determined to be of a score higher than 0.8 (Cohen, 1960). A sample of 50 queries from the total of 1988 queries was used where two other researchers were tasked to annotate each query. The findings of these two researchers were compared whereby the author had a Kappa score of 0.879 with researcher A and a Kappa score of 0.880 with researcher B. The researchers had a Kappa score of 0.880.

Generate Category Features

In the dataset, each row represents one participant. Each participant answers 8 out of 16 queries, of which one is discarded since that is a tutorial. These 16 queries are based on 8 questions, so each question generates two queries. This is because each question can produce a ”correct” query or an ”incorrect” query. This is the first axis. The second axis is the SQL category. As defined by the Spider dataset (Easy, Medium, Hard, Extra-hard), every SQL category is evenly covered by these queries. This means for every category, there are four queries. The implementation of queries ensures that each participant covers at least each category once, often twice. Consequently, the 16 queries are based on three factors: four SQL categories, two questions per category, and two options per question.

4.6 Results

4.6.1 Descriptive Statistics

After data preparation, out of the 476 submissions, only 303 submissions were considered for this section. Randomly the participant was assigned either to the Baseline ($N_{Baseline} = 161$) or to use Color-coding ($N_{color\_coding} = 142$). However, given the extensive use of ANOVA, the groups are balanced to be $N = 142$. This balancing is performed by randomly sampling the Baseline down to 142.

Figure 4.7 visually shows the relation between time using IRA and the number of queries correctly answered. None of the participants had a score of 7. Most of the participants had a score of 2 (N = 74). Figure 4.7 shows Baseline and Color-coding differed most for a score of 2, compared to the other bar plots.

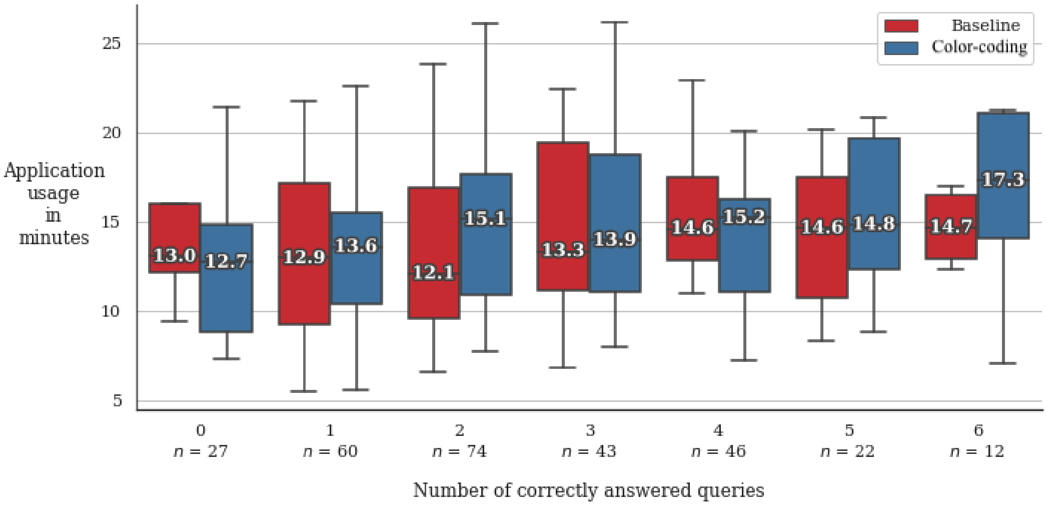


Figure 4.7: Participants' time in minutes using IRA w.r.t. the number of correctly answered queries, Study 2.

The average time spend on IRA was 14 minutes and 59 seconds (±5 minutes, 56 seconds, N = 284), while the median is 13 minutes, 55 seconds as shown in Figure 4.8. Comparatively, when making a distinction between Color-coding ($\bar{x}$ = 15 minutes, 5 seconds, ±5 minutes, 47 seconds, Mdn = 14 minutes, 19 seconds) and Baseline ($\bar{x}$ = 14 minutes, 53 seconds, ±6 minutes, 6 seconds, Mdn = 13 minutes, 19 seconds) we observe that they differ slightly.

The participants were asked two open questions (OQ). The first was part of the pre-survey (OQ4). OQ4 was used to identify if participants that identify with a digital rather than analog example would perform better since they might be more familiar with digital. The outcome is described in Section 4.7.8.

The last question (OQ5) allowed participants to leave their comments and suggestions.

- **OQ4: "*Imagine visualizing a large collection of information containing many cat-***

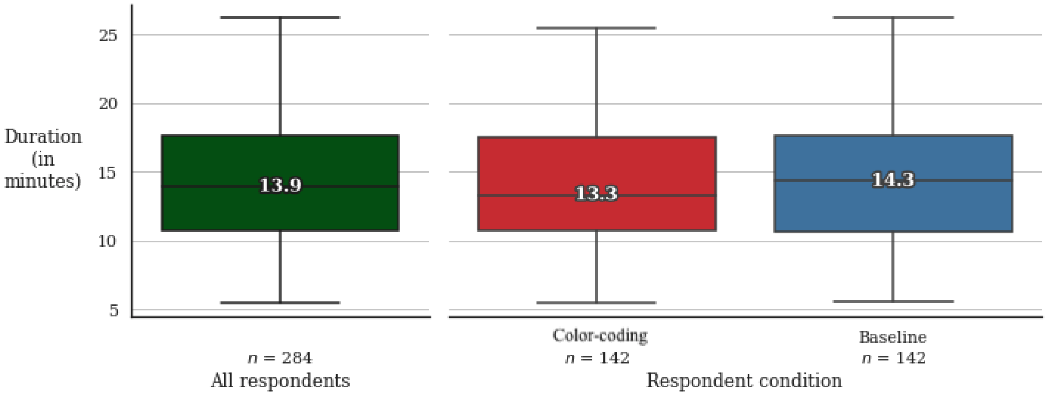


Figure 4.8: Participants' time in minutes using IRA, Study 2.

***egories. Please describe in at least 10+ words how the concept of 'a large collection of information, containing many categories' in your mind could best be visualized. Feel free to use examples you are most familiar with*"**: 284 responses.

- **OQ5: "*Please leave a comment if you have any questions, suggestions, or difficulty you experienced while using the application*"**: 88 responses.

OQ4 The results show that most participants think of digital concepts rather than analog. The most common examples are a Spreadsheet program (24.3%), File explorer (11.5%), and Visualization program (11%). These are primarily digital-only concepts. Analog only concepts were Filing cabinets (4.8%, spot 10). In total, 52.4% accounted for digital concepts, 24.4% for mixed, 16.8% for analog, and 6.1% were unable to be linked to any of these categories.

OQ5 Most participants who provided feedback seemed to agree that they experienced difficulty using IRA. The top 3 are Difficult (19.5%), Clunky layout (17.3%), and Instructions unclear (11.3%). Some positive feedback categories are Easy to use (9%) and Liked (6.8%). 67% of the comments were negative, 15.8% positive and 17.2% were neutral or related to feature requests.

Potential Biases

Often many kinds of potential biases can be observed when dealing with crowd-sourced data (Draws et al., 2021). In Table 4.3 we provide six potential biases in combination with examples found in the dataset.

4.6.2 Hypothesis Tests

We define multiple categories for the hypotheses tests. Each of these categories is covered by Table 4.4. Our hypotheses investigate the relation between the independent variable,

Potential Biases	Dataset Answer Example	Explanation
Automation bias	*Yes, I would assume the computer knows how to calculate averages.*	The response indicates an over-reliance on the decision support system, while it is without merit.
Response bias	*yes because as far as I can tell the ages were added up and divided by the whole number of dogs and the number looks very precise.*	The response is based on the premise that since the answer is exact (5.06...7), it must be correct.
Decoy effect	*no, because the phone number is wrong for Taryn*	The response identifies an unusual pattern in the database (e.g. (275)939-2435x80863). An x in a telephone number is unusual and thus acts as a decoy for the real problem.
Illusory correlation	*yes because 7 people were seen by actual qualified veternarians*	The response incorrectly perceives a relationship between the expected answer of 7 and the number of veterinarians (also 7) classified as professionals.
Self-interest bias	*Yes this is likely to be correct. This is because that number seems right to me when I estimate the data myself, not because I took the time to check it.*	The response indicates that the participant only took the minimally required amount of effort for the task to save time.
Confirmation bias	*Yes, because the number of vets/professionals say 7 while the rest are employees and only vets gives treatments*	The response qualifies employees not as professionals while veterinarians are identified as such. This does not seem right since both employees and veterinarians are found in the professionals' table.

Table 4.3: Examples of potential biases in the dataset of Study 2.

called condition (Color-coding and Baseline), SQL category (Easy, Medium, Hard, Extra-hard), and their combination. Their respective dependent variables (time in seconds and score) are tested separately. The variables' main and interaction effects are measured, requiring a double 2 x 4 (two-way) ANOVA.

For the double 2 x 4 ANOVA, the dependent variables are continuous, while the independent variables consist of two or more categorical independent groups. The results are summarized in Table 4.7.

For Hypothesis 21 a one-way ANOVA was conducted to determine if the average score was different per condition (Color-coding or Baseline). Participants were classified into

		SQL Category				
Condition	**Metric**	**Easy**	**Medium**	**Hard**	**Extra-hard**	**Total**
Baseline	score	0.29 (±0.37)	0.44 (±0.42)	0.39 (±0.39)	0.31 (±0.39)	0.36 (±0.23)
	time (s)	78.43 (±46.42)	110.96 (±74.5)	120.61 (±143.98)	116.65 (±61.16)	105.9 (±50.44)
Color-coding	score	0.29 (±0.36)	**0.51 (±0.42)**	0.33 (±0.38)	0.29 (±0.38)	0.35 (±0.22)
	time (s)	88.57 (±67.65)	**106.49 (±66.55)**	120.93 (±65.37)	**113.17 (±60.23)**	107.24 (±47.94)
Total	score	0.29 (±0.36)	0.48 (±0.42)	0.36 (±0.39)	0.3 (±0.38)	0.35 (±0.23)
	time (s)	83.5 (±58.13)	108.72 (±70.55)	120.77 (±111.61)	114.91 (±60.61)	106.57 (±49.13)

Table 4.4: Color-coding used vs. SQL category results, Study 2.

two groups: Color-coding (N = 142) and Baseline (N = 142). The average score for Color-coding ($\bar{x}$ = 0.35, σ = 0.22) was lower than Baseline ($\bar{x}$ = 0.36, σ = 0.23). However, the differences between these conditions were not statistically significant, F(1) = 0.036, p = 0.849, partial n^2 = 0.006.

For Hypothesis 22 a one-way ANOVA was conducted to determine whether the average time in seconds was different per condition (Color-coding or Baseline). Participants were classified into two groups: Color-coding (N = 142) and Baseline (N = 142). The average time in seconds for Color-coding ($\bar{x}$ = 107.24, σ = 79.78) was higher than Baseline ($\bar{x}$ = 105.9, σ = 90.04). However, the differences between these conditions were not statistically significant, F(1) = 0.018, p = 0.893, partial n^2 = 111.877.

For Hypothesis 23 a two-way ANOVA was conducted to determine if the average score was different per condition (Color-coding or Baseline) and SQL category (Easy, Medium, Hard, Extra-hard). Participants were classified into two groups: Color-coding (N = 142) and Baseline (N = 142) where each participant covered every SQL category: Easy (N = 142), Medium (N = 142), Hard (N = 142) and Extra-hard (N = 142). The interaction effect between condition and SQL category was not statistically significant, F(3) = 1.564, p = 0.196, partial n^2 = 0.708. A follow-up analysis on the main effect for condition (H21) and SQL category (H25) was performed.

For Hypothesis 24 a two-way ANOVA was conducted to determine if the average time in seconds was different per condition (Color-coding or Baseline) and SQL category (Easy, Medium, Hard, Extra-hard). Participants were classified into two groups: Color-coding (N = 142) and Baseline (N = 142) where each participant covered every SQL category: Easy (N = 142), Medium (N = 142), Hard (N = 142) and Extra-hard (N = 142). The interaction effect between condition and SQL category was not statistically significant, F(3) = 0.515, p = .672, partial n^2 = 9475.909. A follow-up analysis on the main effect for condition (H22) and SQL category (H26) was performed.

For Hypothesis 25 a one-way ANOVA was conducted to determine whether the average score was different per SQL category (Easy, Medium, Hard, Extra-hard). Participants were classified into four groups: Easy (N = 142), Medium (N = 142), Hard (N = 142) and Extra-hard (N = 142). The average score for Easy ($\bar{x}$ = 0.29, σ = 0.36) was lowest, followed by Extra-hard ($\bar{x}$ = 0.30, σ = 0.38), Hard ($\bar{x}$ = 0.36, σ = 0.39) and then Medium ($\bar{x}$ = 0.48, σ = 0.42). The differences between the SQL categories were statistically significant, F(3) = 14.362, p <.001, partial n^2 = 6.504.

Tukey HSD post hoc analysis were performed with a 95% confidence interval as shown in Table 4.5. SQL category Easy (MD 0.192, CI [0.108, 0.276], p <0.001), Hard (.146, CI 0.118, CI [0.034, 0.202], p = 0.002) and Extra-hard (MD 0.178, CI [0.094, 0.262], p <0.001) had a statistically significantly lower mean score than Medium. Other combinations revealed no such finding.

group 1	**group 2**	**Statistic**	p	**Reject?**
Easy	Medium	MD = 0.192, $C.I.$ = [0.108, 0.276]	<0.001	**True**
Extra-hard	Medium	MD = 0.178, $C.I.$ = [0.094, 0.262]	<0.001	**True**
Hard	Medium	MD = 0.118, $C.I.$ = [0.034, 0.202]	0.002	**True**
Easy	Hard	MD = 0.074, $C.I.$ = [-0.01, 0.158]	0.106	False
Extra-hard	Hard	MD = 0.06, $C.I.$ = [-0.024, 0.144]	0.257	False
Easy	Extra-hard	MD = 0.014, $C.I.$ = [-0.07, 0.098]	0.973	False

Table 4.5: Pairwise Tukey-HSD post-hoc test for SQL category w.r.t. scores for H25, Study 2.

For Hypothesis 26 a one-way ANOVA was conducted to determine if the average time in seconds was different per SQL category (Easy, Medium, Hard, Extra-hard). Participants were classified into four groups: Easy (N = 142), Medium (N = 142), Hard (N = 142) and Extra-hard (N = 142). The average time in seconds for Easy ($\bar{x}$ = 83.5, σ = 58.13) was lowest, followed by Medium ($\bar{x}$ = 108.72, σ = 70.55), Extra-hard ($\bar{x}$ = 114.91, σ = 60.61) and then Hard ($\bar{x}$ = 120.77, σ = 111.61). The differences between the SQL categories were statistically significant, F(3) = 12.457, p <.001, partial n^2 = 229276.032.

Tukey HSD post hoc analysis as shown in Table 4.6, were performed with a 95% confidence interval. SQL category Medium (MD 25.222, CI [8.328, 42.115], p = 0.001), Hard (MD 37.266, CI [20.372, 54.159], p <0.001) and Extra-hard (MD 31.412, CI = [14.518, 48.305], p <0.001) had a statistically significantly higher mean time in seconds than Easy.

group1	**group2**	**Statistic**	p	**Reject?**
Easy	Hard	MD = 37.266, $C.I.$ = [20.372, 54.159]	<0.001	**True**
Easy	Extra-hard	MD = 31.412, $C.I.$ = [14.518, 48.305]	<0.001	**True**
Easy	Medium	MD = 25.222, $C.I.$ = [8.328, 42.115]	0.001	**True**
Hard	Medium	MD = -12.044, $C.I.$ = [-28.938, 4.849]	0.258	False
Extra-hard	Medium	MD = -6.19, $C.I.$ = [-23.084, 10.703]	0.782	False
Extra-hard	Hard	MD = 5.854, $C.I.$ = [-11.04, 22.747]	0.809	False

Table 4.6: Pairwise Tukey-HSD post-hoc test for SQL category w.r.t. time spent in seconds for H26, Study 2.

#	**Hypothesis**	**Statistic**	p	α	**Reject?**
H21	There is no difference in score between Color-coding vs. Baseline.	$F = 0.036$, $df = 1$, $n^2 = 0.006$	0.849	0.02500	False
H22	There is no difference in time between Color-coding vs. Baseline.	$F = 0.018$, $df = 1$, $n^2 = 111.877$	0.893	0.05000	False
H23	There is no interaction difference in score between (Color-coding vs. Baseline) and SQL categories.	$F = 1.564$, $df = 3$, $n^2 = 0.708$	0.196	0.01250	False
H24	There is no interaction difference in time between (Color-coding vs. Baseline) and SQL categories.	$F = 0.515$, $df = 3$, $n^2 = 9475.909$	0.672	0.01667	False
H25	There is no difference in score between the SQL categories.	$F = 14.362$, $df = 3$, $n^2 = 6.504$	0.001	0.00833	**True**
H26	There is no difference in time between the SQL categories.	$F = 12.457$, $df = 3$, $n^2 = 229276.032$	0.001	0.01000	**True**

Table 4.7: A summary of the performed ANOVA tests for Study 2. These were corrected using Holm-Bonferonni. The null hypotheses could be rejected of H25 and H26.

4.7 Discussion

Two out of six of our hypotheses were rejected. These were related to the SQL difficulties and therefore not related to conditions Color-coding and Baseline or a combination of conditions with SQL difficulties. Below we discuss the results that led us to these findings, their implications, and their limitations.

4.7.1 Statistical Significant findings

The statistically significant differences can be explained by delving deeper into the different covariates present in the data. Table 4.8 displays the performance with regards to queries that were part of the survey. The survey has a total of sixteen queries, of which half expect the participant to identify as wrong and the other half as correct.

Each participant gets eight of these sixteen queries randomly assigned of which one is a tutorial. So in an ideal situation, the participant evaluates half of these queries as *wrong* while expecting the other half to be considered *correct*. Table 4.8 shows that participants were more likely to correctly evaluate queries to be *wrong* using less time compared to queries that are expected to be evaluated *correct*. We suppose this is because it is easier to reason why something is wrong rather than right.

Expected answer per SQL difficulty	**Baseline**	**Color-coding**
Easy - wrong (score)	0.24 (±0.4)	0.23 (±0.38)
Easy - wrong (time (s))	87.52 (±66.23)	95.71 (±112.35)
Medium - wrong (score)	0.7 (±0.44)	**0.77 (±0.41)**
Medium - wrong (time (s))	92.53 (±59.59)	**80.3 (±62.78)**
Hard - wrong (score)	0.58 (±0.48)	0.58 (±0.48)
Hard - wrong (time (s))	93.54 (±64.09)	112.99 (±82.2)
Extra-hard - wrong (score)	0.48 (±0.49)	0.36 (±0.46)
Extra-hard - wrong (time (s))	110.79 (±56.5)	**109.57 (±76.73)**
Easy - correct (score)	0.33 (±0.44)	0.32 (±0.44)
Easy - correct (time (s))	72.97 (±49.98)	80.64 (±54.83)
Medium - correct (score)	0.21 (±0.4)	**0.24 (±0.4)**
Medium - correct (time (s))	128.58 (±99.86)	**128.08 (±77.82)**
Hard - correct (score)	0.24 (±0.4)	0.19 (±0.37)
Hard - correct (time (s))	132.01 (±154.61)	**128.95 (±74.45)**
Extra-hard - correct (score)	0.2 (±0.38)	**0.25 (±0.42)**
Extra-hard - correct (time (s))	120.49 (±79.87)	**115.16 (±75.69)**

Table 4.8: Conditions vs. SQL categories w.r.t. expected answer for query, Study 2.

In Table 4.9 queries are categorized according to the uncontrolled variables from Section 4.3.5. During the survey every participant answers eight out of sixteen possible queries by answering either by affirmation or negation and their respective reasoning. Since these participants only cover eight out of sixteen queries it means that a participant does not cover every (uncontrolled) category, which can be observed in Table B.1. Thus the minimal sample size for each of these categories fluctuates and is at least 56 for each condition (Baseline and Color-coding).

Table 4.4 shows that for every category, except Medium, the performance was lower for Color-coding compared to Baseline. This difference for category Medium might be due to these queries having only 2 to 3 color-coded columns, while other categories have 1 to 2. This might indicate that color-coding is more useful when more columns are colored. The aforementioned is supported by findings in Table 4.9, where three color-coded columns perform better when using Color-coding compared to when using it for 1 or 2 columns.

Then we observe four different types of errors introduced by the model in Table 4.4. These are adding column, wrong column, missing column and missing where clause. The errors are based on the ablation study of the used RAT-SQL implementation (Wang et al., 2020), which are artificially inserted for half of the queries in the survey.

We see that the category missing columns and adding columns perform better when using Color-coding. We hypothesize that it is easier to observe missing and added colors for participants, which is why it might take less time to answer than Baseline.

Another observation is that the wrong column category performs relatively much worse for Color-coding columns than Baseline. A possible reason might be that answers reasonably close to the expected output are more quickly accepted as correct rather than when it is wrong. An extra column or a wrong column is closer to a correct answer than when a column or where clause is missing. Some might argue that having an additional column is not really a wrong answer and that with a wrong column, you still are presented the right information (e.g. instead of returning the description of a product, you return the id).

We also observe that the category of aggregation operations perform better for Baseline while select queries perform better for Color-coding.

4.7.2 Annotations

In Appendix A the annotation process is described. This is followed by Appendix B, in which the categories of Table B.1 are described. In this table we can observe the query and the categories it incorporates as used in Table 4.9, and also the annotation statistics.

This annotation process provided insight into multiple potential biases present in the responses of the study participants. This was reported in Table 4.3. These observations are discussed below in the analysis.

Annotation analysis

Some queries are comparatively more involved than others. For example this can be because it requires calculation of the average (e.g. query 1, 2, 9 and 10 from Appendix A) or matching of two tables by looking at two different identification numbers (e.g. query 11, 12, 13, 14, 15 and 16 from Appendix A). Such considerations influence the difficulty of the task at hand for a crowdworker and is reflected by the ratio of correct, partially correct, and incorrect (See Table B.1). This adds a layer of complexity to having users of NLIDB systems verify answers provided by the model because some answers are easier to verify than

Query categories	Baseline	Color-coding
Average time spent per question (time (s))	105.9 (±90.04)	107.24 (±79.78)
Average score	0.36 (±0.23)	0.35 (±0.22)
#1 color-coded column (score)	0.32 (±0.3)	0.28 (±0.28)
#1 color-coded column (time (s))	107.13 (±71.23)	**106.31 (±71.59)**
#2 color-coded columns (score)	0.43 (±0.31)	0.41 (±0.33)
#2 color-coded columns (time (s))	106.44 (±113.36)	**109.69 (±90.19)**
#3 color-coded columns (score)	0.33 (±0.47)	**0.42 (±0.5)**
#3 color-coded columns (time (s))	99.21 (±72.95)	103.44 (±78.64)
adding column (score)	0.09 (±0.29)	**0.12 (±0.33)**
adding column (time (s))	97.5 (±65.93)	**88.86 (±50.93)**
wrong column (score)	0.57 (±0.49)	0.44 (±0.48)
wrong column (time (s))	88.51 (±70.34)	111.23 (±129.35)
missing column (score)	0.7 (±0.44)	**0.77 (±0.41)**
missing column (time (s))	92.53 (±59.59)	**80.3 (±62.78)**
missing where clause (score)	0.5 (±0.47)	0.44 (±0.46)
missing where clause (time (s))	103.0 (±56.5)	107.97 (±76.56)
aggregate (score)	0.32 (±0.3)	0.28 (±0.28)
aggregate (time (s))	107.13 (±71.23)	**106.31 (±71.59)**
aggregate (count) (score)	0.35 (±0.35)	0.29 (±0.33)
aggregate (count) (time (s))	111.26 (±73.86)	111.91 (±75.37)
aggregate (avg) (score)	0.23 (±0.42)	0.23 (±0.42)
aggregate (avg) (time (s))	94.17 (±60.73)	**90.45 (±56.95)**
select' (score)	0.4 (±0.29)	**0.42 (±0.29)**
select (time (s))	104.71 (±105.09)	108.19 (±87.52)

Table 4.9: Query categories which were not controlled for during sampling to be balanced. However, each category has $N > 56$, Study 2.

others. It also implies that different types of questions might require different explanation mechanisms to enable NLIDB users to verify answers correctly.

Another consideration is that what is acceptable for the model might not be accaptable for the user and vice versa. An example of this might be query 1 (Appendix A), where the inclusion of dog_id is erroneous from a model perspective because an extra column is included. In contrast, this might not be problematic from a user perspective since it captures all the necessary information. Query 3 introduces a similar problem; it selects the wrong column but points to the correct row of data, which arguably is similar enough to be valid. Another example is the answer "Yes, I would assume the computer knows how to calculate averages". This indicates a belief that what might seem obvious for a participant might not be for a model.

Next, some of the queries (Appendix A) are identified by participants to be ambiguous (e.g., an answer for query 15: "[... t]he question format is ambiguous. [...]"). This makes verifying also less reliable because multiple interpretations are possible. The queries from Study 2 were explicitly selected to be unambiguous by a group of three people. So, ambiguity is expected to be more commonplace in a natural setting and pose challenges for NLIDB systems and their users.

Furthermore, Study 2 only included participants that self-assessed to be unfamiliar with SQL. This means that these participants are less likely to be familiar with the kinds of SQL errors that the NLIDB model introduces. Consequently, a common mistake found in participants' query responses is reporting distinctive patterns found in the database rather than the errors made by the model to select the appropriate data in the appropriate structure from the database.

4.7.3 7 Principles of Universal Design

The comparison made in Figure 4.9 is between participants identified as SQL non-proficient users of Study 1 and all participants from Study 2. The difference in outcome was expected since both studies' setup was different. Study 1 was conceptual, while Study 2 first showed an implementation, whereafter the Principles questions were posed. The first question is what kind of values an NLIDB system should embody; the second, in what order of importance does IRA represent these values.

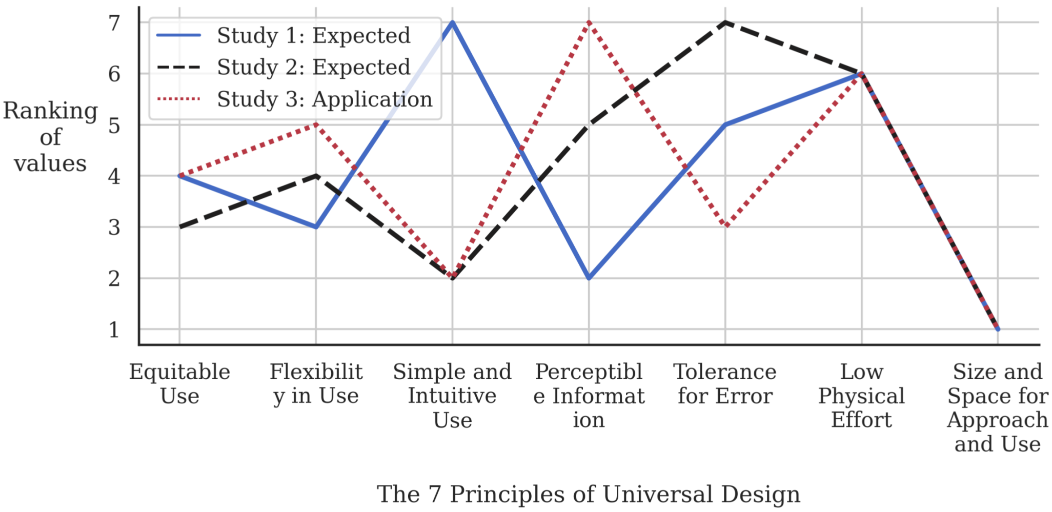


Figure 4.9: The 7 Principles of Universal Design: Ranking of Values of Study 1 and 2.

Regarding ranking, the figure shows the unanimity between the last two values of *"Low Physical Effort"* and *"Size and Space for Approach and Use"*. However, for the other values, there was no such unanimity. We think the interaction with IRA biased the perception of importance of these values. An example is value *"Simple and Intuitive Use"* and *"Perceptible information"*, of which Study 1 is the near opposite of the outcomes of Study 2. Regarding differences between Study 2 Expected and IRA, there seems to be a big difference for *"Tolerance for Error"* and *"Perceptible Information"*. We think this is because of the limitations observed by the IRA implementation. This implementation had no tolerance for error, would not allow answers to be changed during the use of IRA, and would only accept participants' responses if it was in the appropriate format. This ensured data quality but could have been handled differently. Regarding perceptible information, participants would often provide feedback similar to:

- *"It was a bit difficult to follow with the different tabs and try to remember the variables so I could check if the question outcomes were correct or not."*

4.7.4 System Usability Scale

Figure 4.10 shows the difference between Baseline and Color-coding participants. The System Usability Scale Brooke, 1996 score for Baseline is measured at $\bar{x} = 38.82$ (±8.98), Color-coding at $\bar{x} = 40.40$ (±9.32). This small difference is confirmed by the figure, which shows a relatively similar answer for each of the ten SUS questions.

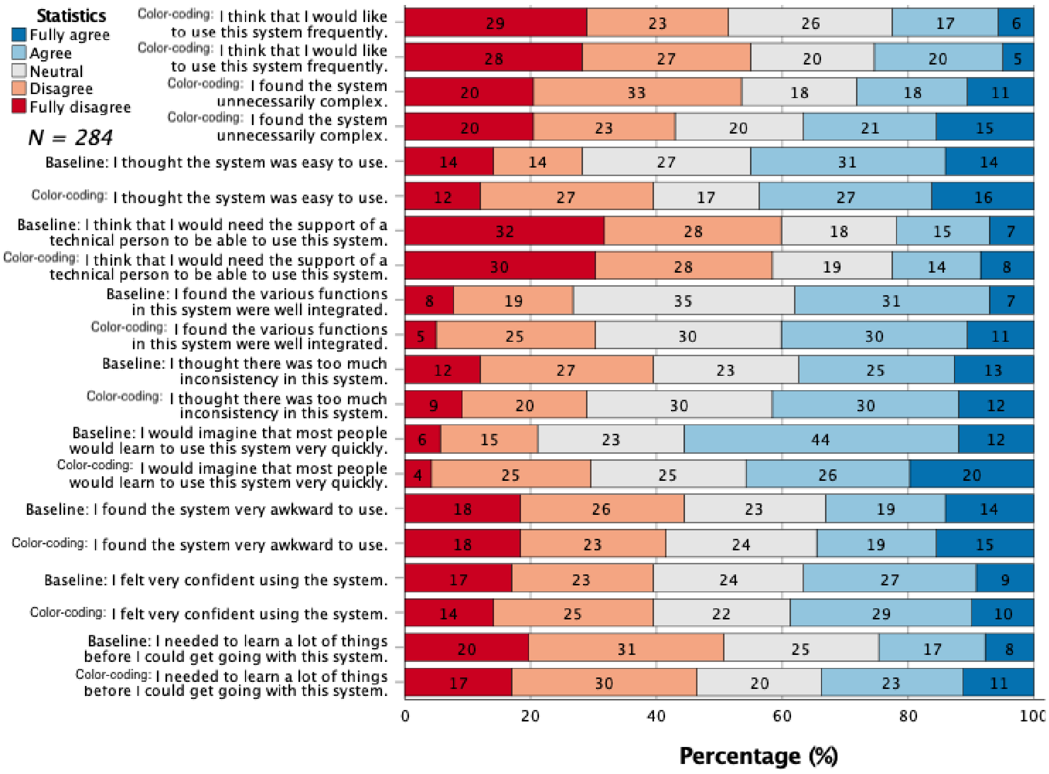


Figure 4.10: System Usability Scale: Difference between Color-coding and Baseline for Study 2.

The questions with the largest differences were questions regarding the topic *"unnecessarily complex"* (Color-coding: $\bar{x} = 2.89$, ±1.37, Baseline: $\bar{x} = 2.65$, ±1.28) and *"learn a lot"* (Color-coding: $\bar{x} = 2.82$, ±1.28, Baseline: $\bar{x} = 2.62$, ±1.20). These were both in favor of the Baseline. We hypothesize that Color-coding made participants more aware of relations that come into play when comparing the data the model used versus the answer it provided. We also observe that Color-coding is slightly higher rated, compared to Baseline.

4.7.5 Survey feedback

The results for OQ5 showed that 33% of the feedback was positive. Out of the 284 participants, only 88 responded with feedback. This means that there was a sizable amount of feedback that should be considered for future works.

We think this experience can be attributed to the fact that the task required the participant to verify a model, which is challenging. Also, the task requires more solid reasoning about why a query is wrong compared to surveys requiring simple preference responses. How hard a participant perceives a question might also make the experience less appealing for the participant and thus evaluate the study more negatively.

As shown by some of the categories by Table 4.9, aggregation of an average and adding a column to queries were categories that performed worse compared to other categories. These categories were based on one query. We believe calculating the mean of some value is a relatively more difficult task compared to the other query types. Also, we think that some participants might not perceive a query with an extra column as erroneous since it contains the expected result. An example of this could be using an asterisk (*) instead of only specifying the relevant columns. Often this is considered acceptable.

Next, participants mentioned that they discovered some of the features (switching table tabs, scrolling sideways, number of shown records on the screen, use of colors indicating the relation between text and column names) too late or not. This was unexpected since the tutorial explained most of these features. Features that were not described were thought to be common knowledge (i.e. scrolling sideways and pagination). Other feedback was aimed at the chat being too big, the sidebar not being helpful, and the table tabs being too small. Feedback like these comments was not discovered during the Pilot study, even though many participants were contacted privately through Prolific, asking for such feedback.

4.7.6 User group limitations

Participants were recruited through the Prolific platform and excluded if they confirmed to be familiar with SQL or mentioned SQL in one of their answers during the study. This selection was based on the assumption of the paper by Narechania et al., 2021, which assumed that assessment of NLIDBs correctness could be challenging for people lacking query language skills.

During study 2, participants are evaluated based on their performance of correctly assessing when the NLIDB model provides the correct and incorrect answers. We consider it likely that participants familiar with SQL will perform better than participants without SQL since they might understand better what kinds of problems might occur when SQL queries are wrong. For example, during the study, participants often mentioned that the source data was erroneous, which was not part of what needed to be assessed, possibly implying their lack of knowledge on what types of errors to expect from an incorrect SQL query.

Also, this study is limited in time and resources. Finding SQL proficient participants is a challenge, as shown by the sample distribution of study 1. For study 2, we wanted a balanced sample for our n-way ANOVA. If we included a difference in user skill level in our analysis, it would change our two-way ANOVA to a three-way ANOVA, which was likely to be unbalanced. This would potentially diminish our statistical power; thus, we decided only to recruit participants that are SQL non-proficient users.

4.7.7 Color-coding colors

IRA, as shown by Figure 4.4, makes use of colors to distinguish relations between parts of the posed question and the related column(s). The color palette consists of contrasting colors, regardless of participants' potential color impediments (Nichols, n.d.). Another way of approaching such a problem would be to use patterns instead or use both. This could increase their discernibility and potentially improve performance.

4.7.8 Analog and Digital concepts

For the first open question of the questionnaire, participants were asked to describe what first comes to mind when thinking of a large collection of information with many categories.

Concepts used	**Analog**		**Digital**		**Both**		**Unknown**	
Condition	**Base-line**	**Color-coding**	**Base-line**	**Color-coding**	**Base-line**	**Color-coding**	**Base-line**	**Color-coding**
N	25	21	73	76	37	36	7	9
Average score	0.39	0.37	0.40	0.39	0.30	0.31	0.18	0.13
Average time spent per question in seconds	110.81	111.07	101.34	104.99	107.01	110.21	130.02	105.38

Table 4.10: Analog, Digital, Mixed and unknown concepts found for answers to OQ4, Study 2.

As pointed out by Table 4.10, participants that only use Digital concepts spend less time answering queries compared to other groups. Regarding the score, we observe relatively small differences between Analog and Digital. The group identified as using concepts that could be assigned to both Digital and Analog were found to perform almost 10% worse. For the group identified as Unknown, we observed a more considerable difference between scores than the rest. This group answered the open question by not using concepts, or it was difficult to assess what was meant. Also, their Baseline time was the highest of all groups. We think the Unknown group scores lower because their answers indicate difficulty with the assignments. We think it makes intuitive sense that participants using Digital concepts are relatively quicker since they supposedly are more familiar with digital tools.

Chapter 5

Conclusion and Future Work

Study 1 was performed to determine if there is a difference between the segmentation of participants identified as **SQL non-proficient users** and **SQL proficient users** for NLIDBs. Therefore we use different requirements as found in the literature. But first, this required an approach to identify such types of user groups, which is described in Section 3.4. An extensive implementation is made that is based on literature (Kim et al., 2020) and makes use of open source packages. This approach allows the automatic evaluation of SQL queries. The automatic evaluation uses approximate semantic equivalence, syntactic equivalence, and related metrics. The semantic and syntactic equivalence scores account for evaluating a participant either as an SQL non-proficient or an SQL proficient user. However, this requires participants to be able to write SQL queries. This is an automated classification process, which classifies participants as SQL non-proficient users if they do not provide such queries. Then, if these participants provide SQL queries, they must have at least one out of five written SQL queries to be evaluated semantically or syntactically equivalent. This is how participants are identified to belong to either one of these groups, which answers **RQ01**.

Then, as previously stated, preferences by participants segmented in either of these two groups for particular requirements are used to determine if there is a difference between this segmentation. These preferences can be observed in Table 4.1. This meant that forty-two such requirements were identified using related literature as described in Section 3.2.

Next, these user groups and their preferences are tested in Section 3.5. For Study 1, two tests used a Chi-square test of independence and a Mann-Whitney U test. Twenty tests are performed, testing each requirement as shown by Table 3.6. Since we have twenty hypotheses tests, a Holm-Bonferonni adjustment is applied. Following this adjustment, no hypotheses are found to be statistically significant. So no statistically significant differences could be observed using the current segmentation of user groups, answering **RQ02**. Therefore we cannot conclude that SQL proficient users differ from SQL non-proficient users regarding preferences for NLIDBs.

Study 2 tries to identify if there are differences between the conditions of Color-coding and Baseline. With Color-coding, the hypothesis stated that participants would be able to perform better as opposed to a Baseline setup.

Consequently, the application IRA was created to test the different conditions and provide a way for participants to interface with an NLIDB. IRA is based on criteria defined in

Study 1.

For **RQ3** and **RQ4**, no significant differences were found between the conditions defined as Color-coding and Baseline and in combination with SQL difficulty categories. However, for **RQ5**, there were significant differences found between the SQL difficulty categories as shown by Table 4.7. These were further investigated in table 4.5 and Table 4.6 through Post-hoc comparisons. There we observed the groups that were identified to be significantly different.

Our findings suggest that these differences can be attributed to uncontrolled variables as described in Table 4.9, such as which type of errors were introduced and what kind of query was used. This means that using color-coding might be worthwhile for non-aggregate select queries containing multiple color-coded columns. This also implies that using Color-coding in the current setup for aggregate operations can be detrimental to the performance. This loses the relational link between the important words of the natural language sentence question and the column it aggregates over.

5.0.1 Future Work

Based on the results and discussion of Study 1 and Study 2, the following directions for future studies are presented.

- More research is needed to establish an empirical standard describing what makes one SQL query more difficult than another w.r.t. model query generation.
- Examining the difference between user groups with different origins warrants further investigation. Supposedly, groups from industry but not from the banking sector might produce different distinguishable results for Study 1.
- More research is needed to differentiate between the skill level of SQL users rather than only SQL non-proficient and SQL proficient users (e.g. SQ: advanced and expert users). This might reveal different preferences between these newly defined groups.
- Different types of errors might require different types of assessment. For example, Study 2 showed that Color-coding only works for non-aggregate multi-column selection queries. Thus further research is needed to determine what might work for other query types (e.g. aggregation queries).
- Additionally, NLIDBs can produce errors that are not recognized as such by its users. This means some results that might be identified as erroneous by the NLIDB might be considered acceptable or correct by its users (e.g. producing an extra column while it contains all the required columns), which warrants more research.
- Lastly, Further research is needed to determine if NLIDB interfaces might need to expose different types of details depending on the needs of its users. It might seem reasonable to show an SQL query for one user, while it might not be for the other.

Bibliography

Abdi, H. (2010). Holm's Sequential Bonferroni Procedure. *Encyclopedia of research design*, 1–8. https://doi.org/10.4135/9781412961288

Affolter, K., Stockinger, K., & Bernstein, A. (2019). A comparative survey of recent natural language interfaces for databases. *The VLDB Journal*, *28*(5), 793–819. https://doi.org/10/ggt69b

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv*. https://doi.org/10.48550/arXiv.1409.0473

Baik, C., Jagadish, H. V., & Li, Y. (2019). Bridging the Semantic Gap with SQL Query Logs in Natural Language Interfaces to Databases. *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 374–385. https://doi.org/10.1109/ICDE.2019.00041

Bogin, B., Berant, J., & Gardner, M. (2019). Representing Schema Structure with Graph Neural Networks for Text-to-SQL Parsing. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4560–4565. https://doi.org/10.18653/v1/P19-1448

Brooke, J. (1996). Sus: A "quick and dirty'usability. *Usability evaluation in industry*, 189–194.

Brysbaert, M. (2019). How many words do we read per minute? A review and meta-analysis of reading rate. *Journal of Memory and Language*, *109*(104047). https://doi.org/10.1016/j.jml.2019.104047

Cai, Y., & Wan, X. (2020). IGSQL: Database Schema Interaction Graph Based Neural Model for Context-Dependent Text-to-SQL Generation. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6903–6912. https://doi.org/10/gj4mkr

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arXiv*. https://doi.org/arXiv:1409.1259

Chu, S., Li, D., Wang, C., Cheung, A., & Suciu, D. (2017). Demonstration of the Cosette Automated SQL Prover. *Proceedings of the 2017 ACM International Conference on Management of Data*, 1591–1594. https://doi.org/10.1145/3035918.3058728

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, *20*(1), 37–46. https://doi.org/10.1177/001316446002000104

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed). Lawrence Erlbaum Associates.

Daniel, F., Kucherbaev, P., Cappiello, C., Benatallah, B., & Allahbakhsh, M. (2018). Quality Control in Crowdsourcing: A Survey of Quality Attributes, Assessment Techniques, and Assurance Actions. *ACM Computing Surveys*, *51*(1), 1–40. https://doi.org/10/gdsk47

Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., & Sen, P. (2020). A Survey of the State of Explainable AI for Natural Language Processing. *arXiv*. https://doi.org/10.48550/arXiv:2010.00711

Dekeyser, S., de Raadt, M., & Lee, T. Y. (2007). Computer assisted assessment of SQL query skills. *Proceedings of the Eighteenth Conference on Australasian Database - Volume 63*, 53–62.

Došilović, F. K., Brčić, M., & Hlupić, N. (2018). Explainable artificial intelligence: A survey. *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 0210–0215. https://doi.org/10/gf2kws

Draws, T., Rieger, A., Inel, O., Gadiraju, U., & Tintarev, N. (2021). A Checklist to Combat Cognitive Biases in Crowdsourcing. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, *9*, 48–59.

Elgohary, A., Hosseini, S., & Hassan Awadallah, A. (2020). Speak to your Parser: Interactive Text-to-SQL with Natural Language Feedback. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2065–2077. https://doi.org/10.18653/v1/2020.acl-main.187

Elgohary, A., Meek, C., Richardson, M., Fourney, A., Ramos, G., & Awadallah, A. H. (2021). NL-EDIT: Correcting Semantic Parse Errors through Natural Language Interaction. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 5599–5610. https://doi.org/10.18653/v1/2021.naacl-main.444

Faul, F., Erdfelder, E., Lang, A.-G., & Buchner, A. (2007). G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior Research Methods*, *39*(2), 175–191. https://doi.org/10.3758/BF03193146

Foster, A., & Ford, N. (2003). Serendipity and information seeking: An empirical study. *Journal of Documentation*, *59*(3), 321–340. https://doi.org/10.1108/00220410310472518

Gadiraju, U., Kawase, R., Dietze, S., & Demartini, G. (2015). Understanding Malicious Behavior in Crowdsourcing Platforms: The Case of Online Surveys. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 1631–1640. https://doi.org/10/ghwphj

Gan, Y., Chen, X., Xie, J., Purver, M., Woodward, J. R., Drake, J., & Zhang, Q. (2021). Natural SQL: Making SQL Easier to Infer from Natural Language Specifications. *arXiv*. https://doi.org/10.48550/arXiv.2109.05153

Gao, J., Galley, M., & Li, L. (2019). Neural Approaches to Conversational AI. *Foundations and Trends® in Information Retrieval*, *13*(2-3), 127–298. https://doi.org/10.1561/1500000074

Grudin, J., & Jacques, R. (2019). Chatbots, Humbots, and the Quest for Artificial General Intelligence. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–11. https://doi.org/10/gf2k5f

Hui, B., Geng, R., Ren, Q., Li, B., Li, Y., Sun, J., Huang, F., Si, L., Zhu, P., & Zhu, X. (2021). Dynamic Hybrid Relation Network for Cross-Domain Context-Dependent Semantic Parsing. *arXiv*. https://doi.org/10.48550/arXiv.2101.01686

Kantere, V. (2016). Query Similarity for Approximate Query Answering. In S. Hartmann & H. Ma (Eds.), *Database and Expert Systems Applications* (pp. 355–367). Springer International Publishing. https://doi.org/10.1007/978-3-319-44406-2_29

Katsogiannis-Meimarakis, G., & Koutrika, G. (2021). A Deep Dive into Deep Learning Approaches for Text-to-SQL Systems. *Proceedings of the 2021 International Conference on Management of Data*, 2846–2851. https://doi.org/10.1145/3448016.3457543

Kim, H., So, B.-H., Han, W.-S., & Lee, H. (2020). Natural language to SQL: Where are we today? *Proceedings of the VLDB Endowment*, *13*(10), 1737–1750. https://doi.org/10/gj4mkn

Li, Y., Chen, B., Liu, Q., Gao, Y., Lou, J.-G., Zhang, Y., & Zhang, D. (2020). “What Do You Mean by That?” A Parser-Independent Interactive Approach for Enhancing Text-to-SQL. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6913–6922. https://doi.org/10/gj4frf

Loyola-González, O. (2019). Black-Box vs. White-Box: Understanding Their Advantages and Weaknesses From a Practical Point of View. *IEEE Access*, *7*, 154096–154113. https://doi.org/10.1109/ACCESS.2019.2949286

Mondal, S., Mukherjee, P., Chakraborty, B., & Bashar, R. (2019). Natural Language Query to NoSQL Generation Using Query-Response Model. *2019 International Conference on Machine Learning and Data Engineering (iCMLDE)*, 85–90. https://doi.org/10.1109/iCMLDE49015.2019.00026

Narechania, A., Fourney, A., Lee, B., & Ramos, G. (2021). DIY: Assessing the Correctness of Natural Language to SQL Systems. *26th International Conference on Intelligent User Interfaces*, 597–607. https://doi.org/10/gj3dzf

Nichols, D. (n.d.). Coloring for Colorblindness
Retrieved March 14, 2022, from http://www.davidmathlogic.com/colorblind/.

Niwattanakul, S., Singthongchai, J., Naenudorn, E., & Wanapu, S. (2013). Using of Jaccard Coefficient for Keywords Similarity. *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS)*, *1*, 5.

Őzcan, F., Quamar, A., Sen, J., Lei, C., & Efthymiou, V. (2020). State of the Art and Open Challenges in Natural Language Interfaces to Data. *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2629–2636. https://doi.org/10/gmdkgf

Parikh, A. P., Wang, X., Gehrmann, S., Dhingra, B., Faruqui, M., Yang, D., & Das, D. (2020). ToTTo: A Controlled Table-To-Text Generation Dataset. *EMNLP*. https://doi.org/10.18653/v1/2020.emnlp-main.89

Prolific. (n.d.). Prolific's Attention and Comprehension Check Policy
Retrieved February 9, 2022, from https://researcher-help.prolific.co/hc/en-gb/ articles/360009223553-Prolific-s-Attention-and-Comprehension-Check-Policy.

Qualtrics. (n.d.). Fraud Detection
Retrieved February 9, 2022, from: https://www.qualtrics.com/support/survey-platform/survey-module/survey-checker/fraud-detection/.

Radlinski, F., & Craswell, N. (2017). A Theoretical Framework for Conversational Search. *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, 117–126. https://doi.org/10/gcpvhg

Renaud, K., & van Biljon, J. (2004). Teaching SQL — Which Pedagogical Horse for This Course? In H. Williams & L. MacKinnon (Eds.), *Key Technologies for Data Management* (pp. 244–256). Springer. https://doi.org/10/bxqhd3

Rubin, O., & Berant, J. (2021). SmBoP: Semi-autoregressive Bottom-up Semantic Parsing. *arXiv*. https://doi.org/10.48550/arXiv.2010.12412

Scholak, T., Schucher, N., & Bahdanau, D. (2021). PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models. *arXiv*. https://doi.org/10.48550/arXiv.2109.05093

Story, M. F., Mueller, J. L., & Mace, R. L. (1998). *The Universal Design File: Designing for People of All Ages and Abilities. Revised Edition*. Center for Universal Design, NC State University.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *arXiv*. https://doi.org/10.48550/arXiv.1706.03762

Wang, B., Shin, R., Liu, X., Polozov, O., & Richardson, M. (2020). RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7567–7578. https://doi.org/10/gj4mjz

Yao, Z., Su, Y., Sun, H., & Yih, W.-t. (2019). Model-based Interactive Semantic Parsing: A Unified Framework and A Text-to-SQL Case Study. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5447–5458. https://doi.org/10.18653/v1/D19-1547

Yao, Z., Tang, Y., Yih, W.-t., Sun, H., & Su, Y. (2020). An Imitation Game for Learning Semantic Parsers from User Interaction. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6883–6902. https://doi.org/10/gj4mdc

Yu, T., Zhang, R., Er, H., Li, S., Xue, E., Pang, B., Lin, X. V., Tan, Y. C., Shi, T., Li, Z., Jiang, Y., Yasunaga, M., Shim, S., Chen, T., Fabbri, A., Li, Z., Chen, L., Zhang, Y., Dixit, S., . . . Radev, D. (2019). CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and* *the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 1962–1979. https://doi.org/10.18653/v1/D19-1204

Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., Ma, J., Li, I., Yao, Q., Roman, S., Zhang, Z., & Radev, D. (2018). Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3911–3921. https://doi.org/10/gj4fvh

Yu, T., Zhang, R., Yasunaga, M., Tan, Y. C., Lin, X. V., Li, S., Er, H., Li, I., Pang, B., Chen, T., Ji, E., Dixit, S., Proctor, D., Shim, S., Kraft, J., Zhang, V., Xiong, C., Socher, R., & Radev, D. (2019). SParC: Cross-Domain Semantic Parsing in Context. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4511–4523. https://doi.org/10/gj4fwd

Zeng, J., Lin, X. V., Xiong, C., Socher, R., Lyu, M. R., King, I., & Hoi, S. C. H. (2020). Photon: A Robust Cross-Domain Text-to-SQL System. *arXiv*. https://doi.org/10.48550/arXiv.2007.15280

Zhong, R., Yu, T., & Klein, D. (2020). Semantic Evaluation for Text-to-SQL with Distilled Test Suites. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 396–411. https://doi.org/10/gkftj7

Zhong, V., Xiong, C., & Socher, R. (2017). Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. *arXiv*. https://doi.org/10.48550/arXiv.1709.00103

Appendix A

Annotation Process Study 2

An annotation procedure is introduced for Study 2 to allow for a more precise evaluation of the open answers given by the participants during the study. This annotation process was verified by sampling 50 queries from the Study 2 dataset and calculating the Kappa score as found in Table A.1.

Annotator	Annotator	Interrater reliability
Annotator #1	Annotator #2	0.879
Author	Annotator #2	0.879
Author	Annotator #1	0.880

Table A.1: Inter-Annotator Agreement Kappa for Study 2

Figure A.1 shows a summary of the complete annotation process.

A.1 Objective

Study 2 is divided into three sections. One of these sections requires the participant to respond via an open answer format. These open answers start with either yes or no and a rationale. This is because the objective of this part of the study is to have the participant look at a given question, the answer generated by the application model for the given question, and the source information on which this answer is based. Then the participant is asked to verify if the model extracted the right information from this source to arrive at the right answer for a given question.

When the response starts with the correct answer and valid reasoning, an answer is correct. For half of the queries, the participants are expected to begin their answer with “yes”; for the other half, the answer should start with “no”. A response is partially correct if either the reasoning or starting answer is wrong. This means an answer is wrong when both reasoning and starting answer is incorrect.

So, the **correctness** of the participant’s answer is determined, but also the classification of the answer into one of four categories. First, we look at what makes reasoning good or bad and when a participant is rejected.

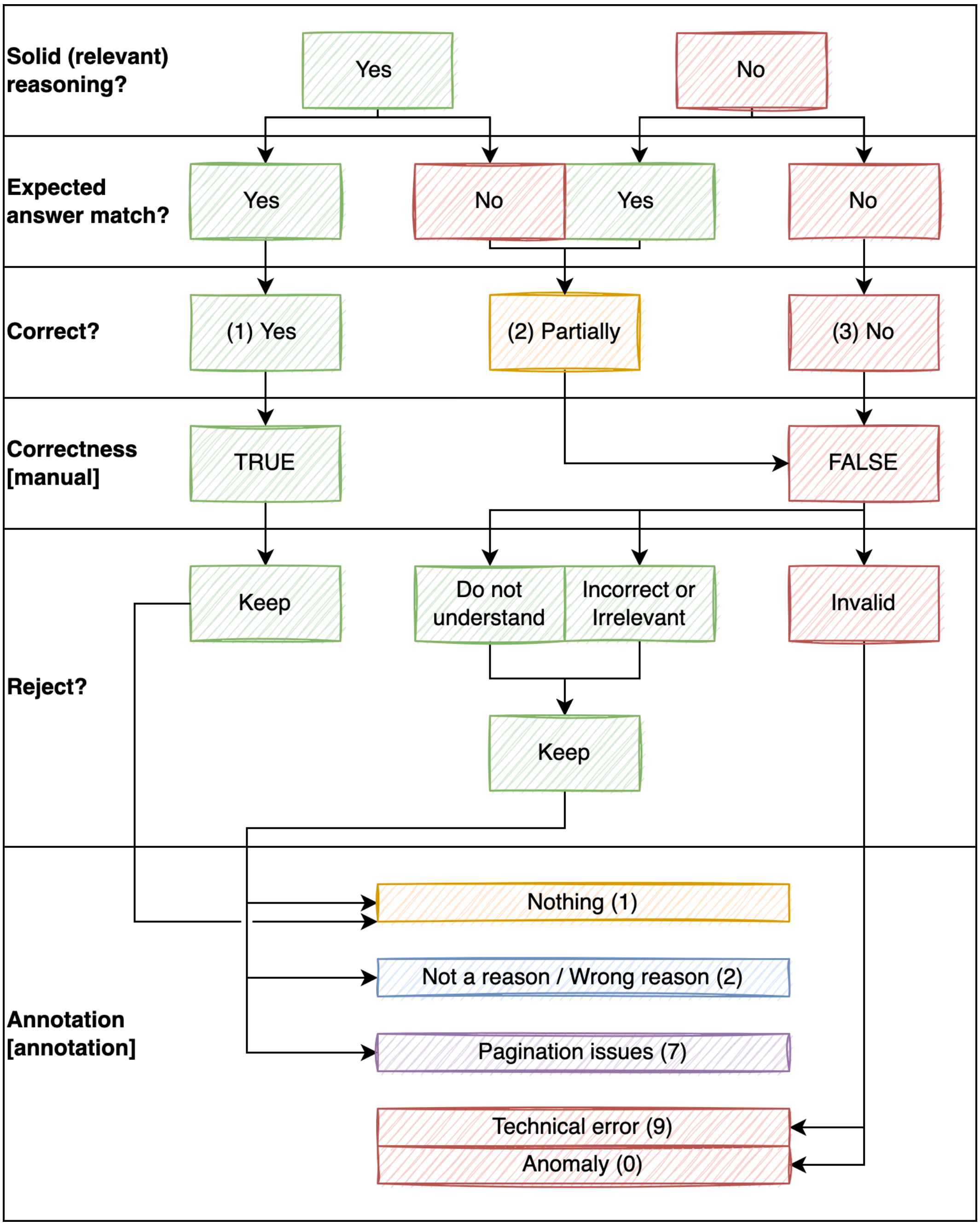


Figure A.1: Query Annotation flow for Study 2.

A.1.1 Bad reasoning

Reasoning can be bad for multiple reasons. In general, this means:

- It indicates the participant does not know how to use the application (e.g. the partici-

pant only sees 5 or 10 records, while the total number per table is 15 records).
- It indicates that the participant does not understand the question (e.g. the participant expects other types of outcomes than are given).
- The reasoning is in the form of “It is correct because it is correct”.
- The reasoning is irrelevant to the answer.
- The reasoning does not reflect the error that was introduced. This only applies to queries that expect answers to start with “No”.

A.1.2 Good reasoning

Reasoning can be good when one of the following applies:

- Queries with expected starting answer “yes”:
  - It indicates that the participant knows how to use the different tables (e.g. mentions that “there are 15 records, of which we only need *x* records”).
  - It indicates that the participant calculated the answer and gives examples. Approximations are acceptable if the query performs column-based operations (e.g. simple SQL select queries, which do not contain where clauses and other types of operations) since those operations do not filter out rows based on values.
- Queries with expected starting answer “no”:
  - It indicates that the participant caught the error and one of the reasons from Queries with expected starting answer “yes”.

A.1.3 Rejection of participants

Some participants are excluded as described in Chapter 4.5. These participants reported technical difficulties (e.g. missing source information) or gave multiple irrelevant answers.

A.1.4 Participant answer annotation

So answers from participants are first annotated to be correct, partially correct, or incorrect. Next, these answers are categorized into one of the five categories. These categories are as follows:

- **Anomaly**: Provides unique reasoning which seems to have little to no merit and cannot be attributed to miscalculations.
- **Nothing**: This indicates no flawed reasoning was found. This often shows the answer was correct or the starting answer was incorrect while the reason was correct.
- **Pagination issues**: The participant is not aware there are more than 5 or 10 records.
- **Wrong reason**: The participant does not understand the task, tables, or data, provides a miscalculation that cannot be attributed to pagination issues, provides a reason which is not a reason, indicates that the participant chose the answer because of preconceived notions or any other kind of bias (e.g. this answer looks correct because it

is too specific to be wrong) or states the answer without why the reason is (in)correct or is just plain wrong.

A.1.5 Dataset intricacies

The queries used for our dataset are based on the database “dog_kennels” from the Spider dataset. This database contains details that influence the answer a participant might give. For example:

- Each table contains either 3 or 15 rows; however, this does not mean each row has unique identification numbers (e.g. some dogs have 0 treatments, others multiple, while there are 15 treatments and 15 dogs).
- There are seven veterinarians and eight employees. These are all identified as professionals.
- There are three different treatment types (e.g. physical examination, vaccination, and take for a walk).
- Some owners have multiple dogs.
- Some home phone and cell numbers might look incorrect.

A.1.6 Application intricacies

Some participants have difficulty interacting with the application. This means that not always the possibility to side-scroll was identified. Also, the application offered the option for the participant to show 5, 10, or 15 records at a time. This led to some participants not knowing this was possible even though the application correctly showed that there were more records available when the option of 5 or 10 records was selected.

A.2 Queries Used

In total, 16 different queries are used for Study 2. Each query covers a different number of categories, either controlled or uncontrolled.

A.2.1 Query 1

Question	What is the average age of all dogs?
Start word	No.
Expected reasoning	The result shows a not relevant extra column . ‘dog identification number’
SQL query	SELECT Avg(Dogs.age), Dogs.dog_id FROM Dogs
Outcome	

Avg(Dogs.age)	dog_id
5.06666666666667	1

A.2.2 Query 2

Question What is the average age of all dogs?

Start word Yes.

Expected reasoning See general guidelines.

SQL query SELECT Avg(Dogs.age) FROM Dogs

Outcome

Avg(Dogs.age)
5.06666666666667

A.2.3 Query 3

Question How much does the most expensive charge type cost?

Start word No.

Expected reasoning The result shows name of the most expensive charge type, not the actual cost.

SQL query SELECT Charges.charge_type FROM Charges ORDER BY Charges.charge_type Desc LIMIT 1

Outcome

charge_type
Health Check

A.2.4 Query 4

Question How much does the most expensive charge type cost?

Start word Yes.

Expected reasoning See general guidelines.

SQL query Charges.charge_amount FROM Charges ORDER BY Charges.charge_amount Desc LIMIT 1

Outcome

charge_amount
640

A.2.5 Query 5

Question What are each owner's first name and their dogs's name?

Start word No.

Expected reasoning The result shows only the owners name. We also want the name of the dog.

SQL query SELECT Owners.first_name FROM Owners JOIN Dogs ON Owners.owner_id = Dogs.owner_id

Outcome

first_name
Jaclyn
⋮
Lorenz

A.2.6 Query 6

Question	What are each owner's first name and their dogs's name?
Start word	Yes.
Expected reasoning	See general guidelines.
SQL query	SELECT Owners.first_name, Dogs.name FROM Owners JOIN Dogs ON Owners.owner_id = Dogs.owner_id
Outcome	

first_name	name
Jaclyn	Kacey
⋮	⋮
Lorenz	Evangeline

A.2.7 Query 7

Question	What are the email, cell phone and home phone of each professional?
Start word	No.
Expected reasoning	the result shows no cell phone number of the professional.
SQL query	SELECT Professionals.email_address, Professionals.home_phone FROM Professionals
Outcome	

email_address	home_phone
deanna.schuster@example.com	+71(6)2898266914
⋮	⋮
jeichmann@example.com	1-138-287-3775

A.2.8 Query 8

Question	What are the email, cell phone and home phone of each professional?
Start word	Yes.
Expected reasoning	See general guidelines.
SQL query	SELECT Professionals.email_address, Professionals.cell_number, Professionals.home_phone FROM Professionals
Outcome	

cell_number	email_address	home_phone
(275)939-2435x80863	deanna.schuster@example.com	+71(6)2898266914
⋮	⋮	⋮
1-258-285-4707x8020	jeichmann@example.com	1-138-287-3775

A.2.9 Query 9

Question How many dogs have an age below the average?

Start word No.

Expected reasoning The result shows the total number of dogs, not number of dogs below average.

SQL query SELECT Count(*) FROM Dogs

Outcome

Count(*)
15

A.2.10 Query 10

Question How many dogs have an age below the average?

Start word Yes.

Expected reasoning See general guidelines.

SQL query SELECT Count(*) FROM Dogs WHERE Dogs.age < (SELECT Avg(Dogs.age) FROM Dogs)

Outcome

Count(*)
9

A.2.11 Query 11

Question Which states have both owners and professionals living there?

Start word No.

Expected reasoning The result shows the identification numbers that overlap for owners and professionals, not the states that overlap. OR we were expecting a list of states not a list of numbers.

SQL query SELECT Owners.owner_id FROM Owners INTERSECT SELECT Professionals.professional_id FROM Professionals

Outcome

owner_id
1
⋮
15

A.2.12 Query 12

Question Which states have both owners and professionals living there?
Start word Yes.
Expected reasoning See general guidelines.
SQL query SELECT Owners.state FROM Owners INTERSECT SELECT Professionals.state FROM Professionals)
Outcome

state
Indiana
Mississippi
Wisconsin

A.2.13 Query 13

Question How many dogs have not gone through any treatment?
Start word No.
Expected reasoning The result shows only the total number of dogs (or treatments), instead of only counting dogs that didn't go through any treatment.
SQL query SELECT Count(*) FROM Dogs
Outcome

Count(*)
15

A.2.14 Query 14

Question How many dogs have not gone through any treatment?
Start word Yes.
Expected reasoning See general guidelines.
SQL query SELECT Count(*) FROM Dogs WHERE Dogs.dog_id NOT IN (SELECT Treatments.dog_id FROM Treatments)
Outcome

Count(*)
6

A.2.15 Query 15

Question How many professionals did not operate any treatment on dogs?
Start word No.
Expected reasoning The result shows the total number of professionals, instead of only counting professionals that did not operate any treatment on dogs.
SQL query SELECT Count(*) FROM Professionals
Outcome

Count(*)
15

A.2.16 Query 16

Question How many professionals did not operate any treatment on dogs?

Start word Yes.

Expected reasoning See general guidelines.

SQL query SELECT Count(*) FROM Professionals WHERE
Professionals.professional_id NOT IN
(SELECT Treatments.professional_id FROM Treatments)

Outcome

Count(*)
7

Appendix B

Annotation Statistics Study 2

Table B.1 describes the statistics related to the 16 queries used in Study 2. These queries each cover different categories. The category condition (Baseline vs. Color-coding) is not considered for these statistics. Accordingly, we find a minimum sample size per query of *N* = 96.

The only controlled category in Table B.1 is **SQL difficulty**, where we cover Easy, Medium, Hard and Extra-hard. The category **# color-coded columns** describes the number of columns that could be colored when color-coding is enabled. Next, for each odd number query, an error is introduced, which is based on errors found in the ablation study of the paper by Wang et al., 2020. The last category is the **SQL operation** used in the SQL. Some operations require more thought from the user when verifying, like calculating an average or a count total, than just selecting all records from a specified column.

Next follows the percentage of correctly answered queries compared to partially correct and incorrect queries. This is manually annotated (according to the process described in Appendix A), using inner-rater agreement as described in Section 4.5.2. Although, for scoring purposes, partially correct are identified as incorrect, making a distinction for analysis is helpful since it makes capturing the uncertainty factor of the question better.

Lastly, there are six annotations identified. These were only considered for answers identified as partially correct and incorrect. We make use of the annotation process as described in Appendix A.

Query	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
SQL difficulty	Easy				Medium				Hard				Extra-hard			
# color-coded columns	1		2				3		1		2		1			
SQL error	Add column	-	Wrong column	-	Missing column	-	Missing column	-	Missing where clause	-	Wrong column	-	Missing where clause	-	Missing where clause	-
SQL operation	average		select						count		select		count			
Correct	0.10	0.33	0.42	0.30	0.85	0.27	0.60	0.20	0.61	0.16	0.58	0.27	0.36	0.16	0.46	0.28
Incorrect	0.44	0.19	0.05	0.08	0.08	0.31	0.36	0.19	0.19	0.45	0.12	0.22	0.21	0.62	0.12	0.46
Partially correct	0.45	0.47	0.53	0.61	0.07	0.42	0.04	0.62	0.20	0.39	0.30	0.52	0.42	0.21	0.42	0.26
Anomaly	0.02	0.01	0.04	0.02	0.11	-	0.02	-	-	-	0.07	-	-	0.02	0.02	-
Nothing	0.04	0.01	0.02	0.01	-	-	0.02	0.02	-	0.02	-	-	-	-	-	-
Pagination issues	0.04	0.08	-	0.01	0.06	-	-	0.02	0.1	0.03	-	0.03	0.06	0.09	0.05	0.02
Wrong reason	0.90	0.90	0.95	0.96	0.83	1.00	0.95	0.96	0.9	0.94	0.93	0.97	0.94	0.89	0.93	0.98

Table B.1: All queries of Study 2 categorized per query category (except Condition: Color-coding vs. Baseline), the correctness ratio and identified annotation types.

Appendix C

Glossary

In this appendix, we provide a list of frequently used terms and abbreviations.

SQL: Structured Query Language; A query language used for relational databases.

NLP: Natural Language Processing; Processing textual data by a computer.

NLIDB: Natural Language Interfaces for Databases; A system that allows the user to interact with a relational database via Natural Language, i.e. ‘plain English’ text.

DL: Deep Learning; A subcategory of Machine Learning, a subcategory of Artificial Intelligence. A computational way of simulating human intelligence via machines.

Spider: The dataset Yu et al., 2018 used for Study 1 and 2.

CoSQL: A dataset based on Spider that includes conversational aspects.

SParC: A dataset based on Spider that includes conversational aspects.

Approximate Semantic Evaluation: ...

NL: Natural language; like English.

OQ#: Open Question.

RQ#: Research Question.

H#: Hypothesis.

SOTA: State of the Art; best that is currently available.

DQL: Data Query Language; a subset of the functionalities available for SQL. It only focuses on querying data, while other subsets might allow data creation and manipulation.

SPA: Single Page Application; Study 2 uses a SPA for the survey.

Non-SQL proficient user: In Study 1, participants who are not evaluated to know SQL are identified as Unskilled users.

SQL proficient user: In Study 1, participants evaluated to know SQL are identified as SQL users.

IRA: Information Retrieval Assistant; The system used by participants during Study 2.

RB: Gadiraju et al., 2015 defines five types of malicious crowd workers. Rule Breakers (RB) do not abide by the instructions of the survey.

GSP: Gadiraju et al., 2015 defines Gold Standard Preys (GSP) as crowd workers who fail gold standard test questions like attention checks.

FD: Another category from a paper by Gadiraju et al., 2015 is Fast Deceivers (FD), which tries to do quick exploitation of the survey.

IE: Ineligible Workers (IE) do not qualify for the survey as defined by Gadiraju et al., 2015.

SD: Smart Deceivers (SD) were not found in Study 1 and 2 because they abide by the instructions but not the intention of the question (Gadiraju et al., 2015).